

**INVESTIGATING SOCIAL NETWORKS WITH
GENALOGICAL FEATURES**

By

Alia S. Alhusban

Supervisor

Dr. Gheith A. Abandah, Prof.

**This Thesis was Submitted in Partial Fulfillment of the Requirements
for the Master's Degree in Computer Engineering and Networks**

**Faculty of Graduate Studies
The University of Jordan**

Dec, 2015

COMMITTEE DECISION

This Thesis/Dissertation (Investigating Social Networks with Genealogical Features) was successfully Defended and Approved on 14-12-2015

Examination Committee Signature

**Dr. Gheith Ali Abandah (Supervisor)
Prof.of Computer Engineering**

**Dr. Andraws Issa Swidan (Member)
Prof. of Computer Engineering**

**Dr. Iyad Fayez Jafar (Member)
Assoc. Prof. of Computer Engineering**

**Dr. Ali Alhaj (Member)
Assoc. Prof. of Computer Engineering
Princess Sumaya University for Technology**

ACKNOWLEDGEMENT

This work would never have been possible without support from people too many to name. However, I would like to thank the following.

I would like to thank my supervisor Prof. Gheith Abandah for giving me the opportunity to work under his guidance. His invaluable technical assistance, moral support, and motivation are the main reasons for the completion of such a challenging study. I would like also to thank the University of Jordan and Computer Engineering Department for giving me the opportunity to pursue my graduate studies.

TABLE OF CONTENTS

| Title | Page |
|---|-------------|
| COMMITTEE DECISION..... | ii |
| ACKNOWLEDGEMENT | iii |
| TABLE OF CONTENTS | iv |
| LIST OF TABLES..... | vii |
| LIST OF FIGURES..... | viii |
| LIST OF ABBREVIATIONS | ix |
| ABSTRACT | x |
| Chapter 1. Introduction | 1 |
| 1.1 Social Networks and Family Trees..... | 1 |
| 1.2 Motivation for Social Networks and Family Trees | 2 |
| 1.3 Problem Statement..... | 2 |
| 1.4 Thesis Contribution | 3 |
| 1.5 Methodology..... | 3 |
| 1.6 Thesis Organization..... | 4 |
| | |
| Chapter 2. Literature Review and Theoretical Background..... | 6 |
| 2.1 Introduction | 6 |
| 2.2 Social Networks..... | 6 |
| 2.2.1 Definition..... | 6 |
| 2.2.2 Social Network Examples | 6 |
| 2.2.2.1 Facebook..... | 6 |
| 2.2.2.2 Twitter | 7 |
| 2.2.2.3 Google+ | 7 |
| 2.3 Genealogical Websites | 7 |
| 2.3.1 Definition..... | 7 |
| 2.3.2 Genealogical Websites Examples..... | 8 |
| 2.3.2.1 Wikitree | 8 |
| 2.3.2.2 Geni.Com..... | 8 |
| 2.3.2.3 Genes Reunited..... | 9 |
| 2.3.2.4 My Heritage..... | 9 |
| 2.3.2.5 Family Link | 10 |
| 2.4 Tree and Tree Algorithms..... | 10 |
| 2.4.1 Tree Data Structure..... | 10 |
| 2.4.2 Tree Terminology | 10 |

| | |
|---|-----------|
| 2.4.3 Type of Trees..... | 11 |
| 2.4.3.1 Binary Tree | 11 |
| 2.4.3.2 None-Binary Tree | 12 |
| 2.4.4 Tree Algorithms..... | 12 |
| 2.4.4.1 Depth-First Algorithm (DFS) | 12 |
| 2.4.4.2 Breadth-First Algorithm | 13 |
| 2.4.4.3 DFS Versus BFS..... | 13 |
| Chapter 3. Requirements Elicitation and Analysis | 15 |
| 3.1 Introduction | 15 |
| 3.2 Questionnaire Description | 15 |
| 3.3 Data Collection | 17 |
| 3.4 Results | 18 |
| 3.5 Requirements | 19 |
| 3.5.1 Functional Requirements | 19 |
| 3.5.2 Non-Functional Requirements..... | 21 |
| 3.6 Use-Case Analysis | 21 |
| 3.6.1 Use-Case Actors | 24 |
| 3.6.2 Use-Case Scenarios | 24 |
| Chapter 4. FG Design | 27 |
| 4.1 Introduction | 27 |
| 4.2 GUI Design..... | 27 |
| 4.2.1 Any User..... | 27 |
| 4.2.2 FG Member..... | 29 |
| 4.2.3 FG Administrator..... | 30 |
| 4.3 Family Tree Algorithm and Implementation..... | 33 |
| 4.3.1 Introduction | 33 |
| 4.3.2 Family Tree Algorithm | 34 |
| 4.3.3 Family Tree Implementation | 35 |
| 4.4 Database Design | 37 |
| Chapter 5. GUI Programming and Tree View Window Programming..... | 41 |
| 5.1 The GUI Programming | 41 |
| 5.1.1 Introduction | 41 |
| 5.1.2 Group Structure | 45 |
| 5.1.3 Wall | 47 |
| 5.1.4 Manage | 48 |

| | |
|--|-----------|
| 5.2 Tree View Window Programming | 48 |
| 5.2.1 User Interaction | 48 |
| 5.2.2 Implementation | 49 |
| 5.3 Functional Requirements Implementation..... | 51 |
| Chapter 6. Requirements Validation and Evaluation | 53 |
| 6.1 Introduction | 53 |
| 6.2 Functionality Validation | 53 |
| 6.2.1 Data Collection | 54 |
| 6.2.2 Results | 55 |
| 6.3 Non-Functional Requirements Validation..... | 56 |
| 6.3.1 Results | 57 |
| 6.3.2 Scalability | 57 |
| Chapter 7. Conclusions and Future work | 58 |
| 7.1 Conclusions | 58 |
| 7.2 Future Work..... | 58 |
| References..... | 60 |
| Appendix A..... | 62 |
| Appendix B..... | 64 |
| Arabic Abstract..... | 66 |

LIST OF TABLES

| NUMBER | TABLE CAPTION | PAGE |
|---------------|---|-------------|
| 1 | SN and Social Services | 1 |
| 2 | BFS and DFS Complexity Measures by Different Languages | 12 |
| 3 | Genealogical Sites and Services | 15 |
| 4 | Questionnaire Results | 19 |
| 5 | Feature Rank and Implementation Decision | 21 |
| 6 | Initial Values for Group Owner | 46 |
| 7 | Initial Root Information for Created Group | 46 |
| 8 | Root Information | 46 |
| 9 | Wall Functions and Descriptions | 47 |
| 10 | Requests Functions and Descriptions | 48 |
| 11 | Questionnaire Results Percentages for Use-Cases | 55 |
| 12 | Questionnaire Results Percentage for None-Functional Requirements | 57 |

LIST OF FIGURES

| NUMBER | FIGURE CAPTION | PAGE |
|--------|---|------|
| 1 | Binary Tree | 11 |
| 2 | Depth-First Pre-Order: F, B, A, D, C, E, G, I, H | 12 |
| 3 | Breadth-First Search Traversal | 13 |
| 4 | Distribution of Respondents Based on Sex, Educational Level, and Age | 17 |
| 5 | Rating of the Suggested Genealogical Features | 18 |
| 6 | Use-Case Diagram | 26 |
| 7 | Any User Screen | 27 |
| 8 | Join FG Screen | 28 |
| 9 | Create FG Screen | 29 |
| 10 | Family Member Screen | 29 |
| 11 | FG Wall Screen | 30 |
| 12 | Request Family Update Screen | 30 |
| 13 | Join Requests Screen | 31 |
| 14 | View Join Request Screen | 31 |
| 15 | Tree Update Screen | 31 |
| 16 | Approve Request Update Screen | 32 |
| 17 | Maintain Family Tree Screen | 33 |
| 18 | Manage Tap Screen | 33 |
| 19 | Family Tree Algorithm | 34 |
| 20 | Database Design | 40 |
| 21 | Sign Up and Login Windows | 41 |
| 22 | User Profile Window | 42 |
| 23 | Edit Profile Information | 43 |
| 24 | Search Group Window | 43 |
| 25 | Create Group Dialog Box | 44 |
| 26 | Any User Group View | 46 |
| 27 | Family Member Group View | 47 |
| 28 | Administrator Group View | 47 |
| 29 | Distribution of Participants Based on Sex, Educational Level, and Age | 55 |
| 30 | Rating of Evaluated FG Use-Cases | 56 |
| 31 | Rating of the Evaluated None-Functional Requirements | 57 |
| 32 | Execution Time for Different Tree Sizes | 57 |

LIST OF ABBREVIATIONS

| | |
|------|----------------------------|
| ASP | Active Server Pages |
| BFS | Breadth First Search |
| DFS | Depth First Search |
| FG | Family Group |
| GUI | Graphic User Interface |
| HTML | Hypertext Mark-up Language |
| ID | Identifier |
| IP | Internet Protocol |
| PHP | Personal Home Pages |
| SN | Social Network |
| SQL | Structured Query Language |

INVESTIGATING SOCIAL NETWORKS WITH GENALOGICAL FEATURES

By

Alia S. Alhusban

Supervisor

Dr. Gheith A. Abandah, Prof

ABSTRACT

Social networking is a service which is provided to people who share interests, activities, and backgrounds. Genealogical services are services provided to relatives and families such as family tree, sharing family events, and accessing family member profiles. Social networks have attracted billions of users, they provide web based services that allow individuals to communicate and share common interests.

Family tree is a special social networking, which has attracted family members to communicate and collaborate. Deploying genealogical features in social networks (SNs) will allow family members to communicate and collaborate and to take the genealogical services and social services benefits. In this thesis, we propose agenda application to connect SN users with their relatives. The application is called Family Group (FG) and it connects family members with each other through the family tree. The main genealogical features are validated through a questionnaire, and the main genealogical features that reflect people requirements are implemented in the FG. The FG requirements, and the tree algorithm used are discussed in details. The FG is validated through some volunteers to test the main functional requirements (percentages of satisfaction); also the tree algorithm scalability is tested to draw large scale family trees in a short period (tree performance).

Chapter 1. Introduction

1.1 Social Networks and Family Trees

Social networks (SN) have attracted billions of users. They provide web based services that allow individuals to communicate and share common interests. SN also includes specialized types, as shown in Table 1. All SNs provide collaborative environments where users can effectively connect and share common interests.

Table 1: SN and Social Services

| SN | Social Services |
|-------------|--|
| Elftown.com | Grouping fans of fantasy and science fiction |
| Ravelry.com | Grouping fans of knitting |
| Flickr | Photo sharing |
| Last FM | Music listening |
| YouTube | Video sharing |

Another type of SN is a family tree, a chart representing family relationships in a tree structure (Wikipedia, 2002). There are many web-based applications for this purpose and many of them can connect family members in social environments. Geni.com is a type of genealogical and social environment which has nearly one million users (Geni, 2011). Its primary focus is connecting families and allowing users to easily create family trees and inviting other family members to join. Each individual in the tree has a profile. Family members can work together to build profiles for their common ancestors.

1.2 Motivation for SN and Family Trees

The importance of the family tree comes from the interest of millions of people in keeping their heritage from a loss over the years. There are many genealogical software applications that build family trees automatically based on the family's database. They can be fed by users themselves, but they usually must be installed on the PC, besides, there are many web based applications where users can access such software, build their family trees online, and share family trees into one family ancestor.

The SN sites have become the most collaborative communication mechanism where members share their activities with each other, and family members can communicate with each other and share photos and news. They attract billions of users, providing communication between people who have common interests.

Current SNs support family history and genealogy through family groups, which are built by the users themselves. Also a user can identify his close relatives on friend list to get notified about their recent activities. Adding genealogical features to a SN site will result in a wide family tree that can help SN users to find their lost relatives. It can also provide a large database that helps genealogist in their research.

1.3 Problem Statement

SN large capabilities and distributed nature can offer perfect environment to provide genealogical services. However, to provide genealogical services in SN, a general application that offers genealogical features in SN is required.

SNs help users to communicate and collaborate regardless of their relations, and they do not provide extensive genealogical features. Genealogical sites help only family members to collaborate. A general system for providing genealogical features in SN

will offer the opportunity for SN users to use genealogical services with their relatives and collaborate with other friends in SN.

1.4 Thesis Contribution

The thesis studies the main genealogical features that meet peoples' requirements. It introduces a system in SN to support the main wanted genealogical requirements. We define the functional and non-functional requirements of our system. Based on peoples' genealogical requirements, we present the system in Facebook network, to link SN users with their family tree. The system is called Family Group (FG). We also design and build FG database that can meet the identified features and requirements.

FG supports the basic group services such as joining and leaving group. It also supports the basic tree operations such as inserting and deleting nodes.

FG provides an interface to link between *group* services and family tree services. FG is a component that can be added to Facebook to provide genealogical services and social services at the same time. At the end, FG is evaluated to test the main requirements.

1.5 Methodology

The methodology of this study is as follows:

- **Investigation of wanted and popular genealogical features:** In this phase, we prepare a questionnaire about the genealogical features that meet people requirements and survey the most popular and wanted genealogical features.
- **Investigation of the current SN and genealogical social sites:** In this phase, we survey the current SN sites and current genealogy SN sites.

- **Design the database and chose family tree algorithm:** Based on features and requirements, we get from the previous phase; we design database and use a family tree algorithm that can meet the identified features and requirements.
- **Building the database and implementing algorithms:** Build the database using suitable language and implement family tree algorithms based on requirements.
- **Design and implement the GUI:** Design and implement friendly user GUI using suitable web programming language.
- **Evaluation and testing:** After completing the site, the site is validated by some volunteers to test the main requirements. The family tree algorithm is also evaluated to test its performance.
- **Documentation and dissemination:** The work is documented and its results are submitted for release in peer-reviewed international conferences/journals.

1.6 Thesis Organization

Chapter 1. The Introduction is concerned with describing briefly the whole idea that stands behind our thesis and its objectives.

Chapter 2. Literature Review and Theoretical Background describe basically the SN and genealogical websites, some examples for both of them, and useful tree algorithms.

Chapter 3. Requirements Elicitation and Analysis define the functional and non-functional requirements of our system. Based on peoples' genealogical requirements, the features are prioritized and illustrated in a use case model.

Chapter 4. Design is concerned with studying tree algorithms, and selecting an efficient algorithm to implement family tree, describing the graphical user interfaces based on the client side perspective. It is also concerned with describing the second development stages: database design.

Chapter 5. GUI Programming and Tree View Window Programming: Describes tree view programming and GUI programming.

Chapter 6. Evaluation and Validation contain the validation of functional requirements and non-functional requirements; it also contains the evaluation of used tree algorithm.

Chapter 7. Conclusion and Future Work contain the thesis results and conclusions and outlines future works that can be made in this field.

Chapter 2. Literature Review and Theoretical Background

2.1 Introduction

In this chapter, we introduce Social Network (SN), SN websites, and genealogical websites. In section 2.4, we discuss tree and some tree algorithms.

2.2 Social Network

2.2.1 Definition

Social Network (SN) is a service which is provided to people who share interests, activities, and backgrounds. Also SN service consists of a representation of each user (often a profile), his or her social links, and a variety of additional services.

2.2.2 SN Examples

SN sites are web-based sites that represent each user as a user profile, and maintain a list of users with whom to share connections.

2.2.2.1 Facebook

One of these SN sites is Facebook, created in 2004. By 2007 Facebook was reported to have more than 21 million registered members generating 1.6 billion page views each day (Ellison, 2007).

Facebook is a great way to meet friends and keep up with what they are doing. Once a user adds a friend to his friend list, he will be notified about the activities of the added friend.

The main functional features in Facebook allow the user to add, remove, poke, and subscribe to notifications about friends. Also Facebook supports sharing information through *groups*.

Individuals or companies can create *Pages* which allow fans of an individual, organization, product, service, or concept to join a Facebook fan club page. Pages look

and behave much like a user's private profile, with some significant differences. The other feature that Facebook has is friend finder. Once the user searches his friends by typing their names, Facebook queries its database and finds people, according to related similar academic backgrounds, similar employer, and location.

2.2.2.2 Twitter

Another SN site is a Twitter (Kwak, 2011). The concept of Twitter relies on messaging services, it allows the user to send messages to friends and family members quickly and easily, a user can also follow other users as well. It is easy to have conversations with others. Once a user creates a profile, he can update his status by tweets. In Twitter, the profile can be created per family and all followers are family members, although there is no way to ensure if the followers are really family members or not and what the relations between members.

2.2.2.3 Google+

Google+ is a social networking and identity service. The main feature of Google+ is permitting users to create and organize different interest groups into circles. Each circle can see feeds from other circle, and sharing web pages with other circles. (Kevin Curran, 2012).

2.3 Genealogical Websites

2.3.1 Definition

Genealogical service is a service which is provided to relatives and families, such as family tree, sharing family events and accessing family member's profiles. Genealogical sites are web based sites that provide collaboration and connectivity between family members.

2.3.2 Genealogical Websites Examples

2.3.2.1 Wikitree

Wikitree was introduced in 2008 as a genealogical web based application for communication and collaboration between members. It is free and user friendly for genealogists and non-genealogists. It is a community that creates single shared family tree using traditional genealogical sources and DNA tests. Any user can have a membership and can create unlimited number of profiles and invite many users to join his family tree. The tree can be displayed in three schemes: ancestors, a graphical view and time line. Wikitree can help any guest to find his roots by typing the last name of his family in the search section. It can also support accuracy through improving worldwide tree and fix mistakes through code which is shared between the administrator of the family tree and other members. Also, it can support privacy through protecting anything that family members might not want it to be published. Wikitree supports only English language (Gensoftreviews, 2011).

2.3.2.2 Geni.com

Web 2.0 (2007) introduced geni.com as a social and genealogical website. It has the ability of creating a family tree of common ancestors rather than spending time in creating multiple trees for the same ancestors. Geni.com has more than 60 million users who can create complete family trees and receive notifications about the activities of other family tree members (Geni, 2011). Each family has a forum where family members can post and notify others. Geni.com supports many languages, it has an optimized database which abstracts multiple family trees for the common ancestor into one family tree ancestor. The main drawback of geni.com is that many trees are private and can be accessed only by members. It also requires that the invited family members should be login to be able to upload photos (Find the best, 2012).

2.3.2.3 Genes Reunited

Genes reunited became one of the largest genealogy website with over 13 million members and over 750 million names listed (Sean Dodson, 2004). Genes reunited permits users to build their family trees by posting on the site and investigating which ancestors they share with other members. It collects genealogy information from UK and WALES only. Users can access the records by subscriptions for free and high privilege levels.

2.3.2.4 My Heritage

My Heritage is another genealogy and SN site which provides sharing of photos, organizing the family, building family tree, and searching for ancestors. It is one of the largest genealogy sites in the world that has more than 75 million profiles and more than 27 million family trees. It supports 40 languages (CBS news, 2006).

My Heritage provides many web services to site members. One of them is photo tagging, which is an ability to identify person faces on photos based on family tree members. Once the person is a member in a family tree, he can be tagged automatically in photos that are uploaded by family members. Another web service that is provided by My Heritage is Super search. Users can easily search the historical records about any person by adding the first name and the last name or based on other criteria. Advanced services are introduced in my heritage which is record matching that allows any member to be notified about the similar profiles that are created (My heritage, 2012).

2.3.2.5 Family Link

Paul Allen (2002) developed a new approach for online genealogy. It is a free online family tree component which is known as Family Link. It focuses on connecting genealogy researchers with other genealogy researchers. The user can search other users

based on the city or country to view uploaded photos of that city and the names of Family Link registered genealogists who live there or have experience doing research there. If they are online users, user can Skype them. If they are offline, user can send them a message. The user can also give them permission to view his uploaded family tree so genealogists can offer help and suggestions for him. (About.com Genealogy, 2011).

2.4 Tree and Tree Algorithms

2.4.1 Tree Data Structure

A tree is a non-linear graph that consists of points called nodes and connections called vertices. The tree that does not have any node is called empty tree. A tree consists of a root node and non-root nodes.

2.4.2 Tree Terminology

The following terms are used with trees (Wirth and Niklaus, 1986):

- **Root:** The top node in a tree
- **Parent:** A node that has one child or more
- **Siblings:** Nodes with the same parent
- **Descendant:** A node that is reachable by following the path from the parent to the child
- **Ancestor:** A node that is reachable by following the path from child to parent
- **Leaf:** A node that does not have children
- **Internal node:** A node which has children
- **Degree:** The largest number of connections between a root and its leaf nodes
- **Edge:** Connection between nodes
- **Path:** A sequence of nodes and edges connecting a node with a descendant

- **Level:** The level of a node is defined by $1 +$ the number of connections between the node and the root.
- **Height of tree:** The height of a tree is the number of nodes on the longest path between the root and a leaf
- **Height of node:** The height of a node is the number of nodes on the longest path between that node and a leaf
- **Depth:** The depth of a node is the number of edges from the node to the tree's root node
- **Forest:** A forest is a set of disjoint trees

2.4.3 Types of Trees

There are many types of trees, including binary trees and non-binary tree. In the following subsection, we will discuss these types and tree algorithms.

2.4.3.1 Binary Tree

A binary tree is a tree where all its internal nodes have at most two children each: left child and right child as shown in Figure 1.

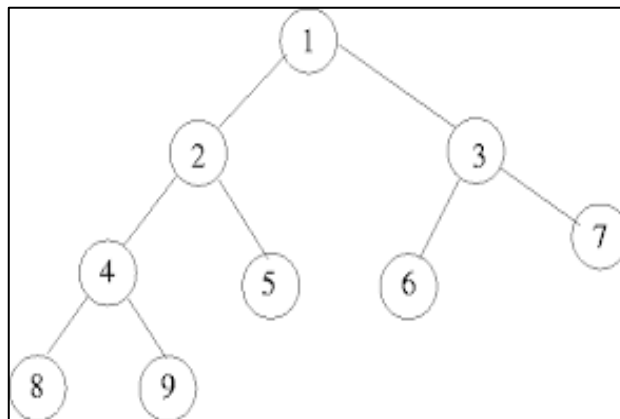


Figure 1: Binary Tree

2.4.3.2 Non-Binary Tree

Non-Binary tree or generic tree is a tree with nodes that have two or more children; each node has numbers of internal sub tree. Because each node in a tree can have a certain number of children, the general tree can be implemented using a first child/next sibling method. Each node has two pointers: one to the leftmost child and one to the rightmost sibling.

2.4.4 Tree Algorithms

There are many algorithms for the order of the nodes visited or how the tree will be drawn. In this section, we discuss these algorithms and a comparison between them.

2.4.4.1 Depth-First Algorithm (DFS)

DFS is an algorithm for traversing or searching a tree. Once it selects the root, it will explore other nodes as far as possible along each branch (Thomas, et al., 2001) as shown in the following figure.

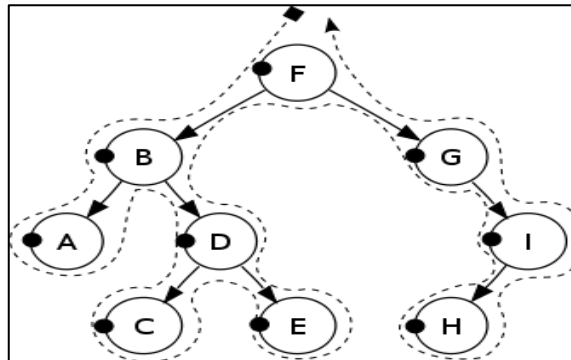


Figure 2: Depth-First Pre-Order: F, B, A, D, C, E, G, I, H

2.4.4.2 Breadth-First Algorithm (BFS)

Trees can be drawn in level-order, where every node is inserted on a level before going to a lower level (Thomas, et al., 2001). Figure 3 shows the node insertion. It starts from the root, moves to level one child, then to level two children, and so on.

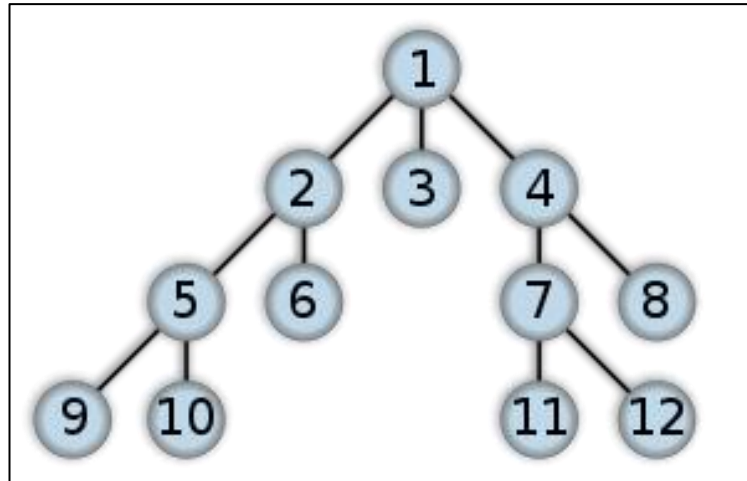


Figure 3: Breadth-First Search Traversal

2.4.4.3 DFS versus BFS

DFS is easier for the programmer to implement because its recursion results that no additional data structures for nodes are needed. On the other hand, BFS needs additional data structures for the nodes that have been drawn.

The two algorithms have been implemented in different programming languages on the same computer (Akanmu, et al., 2010).

Table 1 shows the results for evaluating BFS and DFS algorithms in case of program volume, which is the number of lines in the program, and in the case of program difficulty which is the number of data types in the program.

Table 2: BFS and DFS Complexity Measures by Different Languages

| Language | Program Volume (V) | | Program Difficulty (D) | |
|--------------|----------------------|--------------------|------------------------|--------------------|
| | Breadth-First Search | Depth-First Search | Breadth-First Search | Depth-First Search |
| C | 733 | 491 | 20 | 16 |
| C++ | 998 | 515 | 17 | 18 |
| Pascal | 640 | 539 | 9 | 8 |
| Visual BASIC | 1197 | 1069 | 10 | 9 |

According to Table above, DFS programming complexity is less than BFS, for these reasons, we choose the DFS algorithm to implement family tree.

Chapter 3. Requirements Elicitation and Analysis

3.1 Introduction

The aim of our study is to investigate the most popular genealogical features in SNs. In chapter 2 we survey the most popular genealogical sites and services that provide. The site and genealogical services are summarized in Table 3. We choose some of these services to investigate social network with. Also, users might be interested in some of them; so we make a questionnaire to study the most genealogical features in SN that people desire.

Table 3: Genealogical Sites and Services

| Genealogical sites | Genealogical Features |
|--------------------|--|
| Wikitree | 1-Family tree 2-Accuracy 3-Privcy |
| Geni.com | 1-Family gathering 2-Wall 3-Family tree |
| Genus reunited | 1-Family tree |
| My Heritage | 1-Photo tagging 2-Serach family member 3-Family notification |
| Family Link | 1-Connecting genealogy researchers |

3.2 Questionnaire Description

The questionnaire (Appendix A) contains nine suggested genealogical features. Some of them are deduced from genealogical services as showing in Table 3. The questionnaire investigates the people views about these features. It also investigates the degree of

importance of these features according to people requirements. The questionnaire contains the following genealogical features:

- **Family tree:** The user can build his family tree by using the other SN user information. The user can also invite his relatives to join the tree.
- **Knowledge of the family roots:** This feature gives the user the ability to know his roots by accessing his family tree.
- **Accessing the family members:** After the user becomes a part of the family tree, any family tree member can access his user account and invite his family members to be in his friends list.
- **Separation of friends list and relatives list:** SN should be able to identify easily the user relatives list and family list. By accessing the family tree, the user relative will be identified directly.
- **Relative notifications:** When the user accesses his account, he will receive notifications about the important events and activities that are carried out by his relatives. The order of the notification will be based on the degree of relation.
- **Relative inbox:** By using more secure ways in relative inbox, the relative messages will have privacy and security assurance.
- **Families gathering:** The SN will gather the information about families that have similar roots to make a united family tree.
- **Relative verification:** There might be fake user accounts on the SN site which have the same family name, so these accounts should be verified before adding them in the family tree.
- **Collaborative family tree:** Each family tree has a wall where the members can share their activities and share information about their family.

3.3 Data Collection

The questionnaire is distributed randomly among 82 persons (16 males and 66 females). The data were collected and summarized, according to the age, and education level. Figure 4 shows the distribution of responses based on three categories: sex, age, and education.

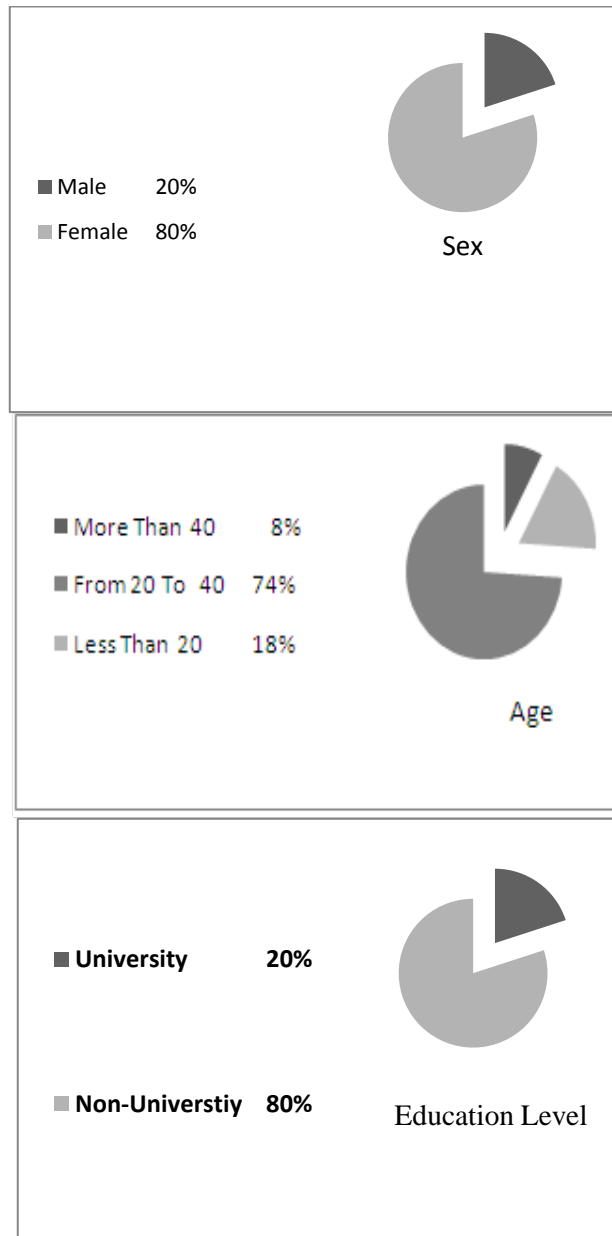


Figure 4: Distribution of Respondents Based on Sex, Educational Level, and Age

3.4 Results

According to the questionnaire responses, the results percentage scores are computed for each genealogical feature based on equation 1. Figure 5 and Table 4 show score percentages for each genealogical feature.

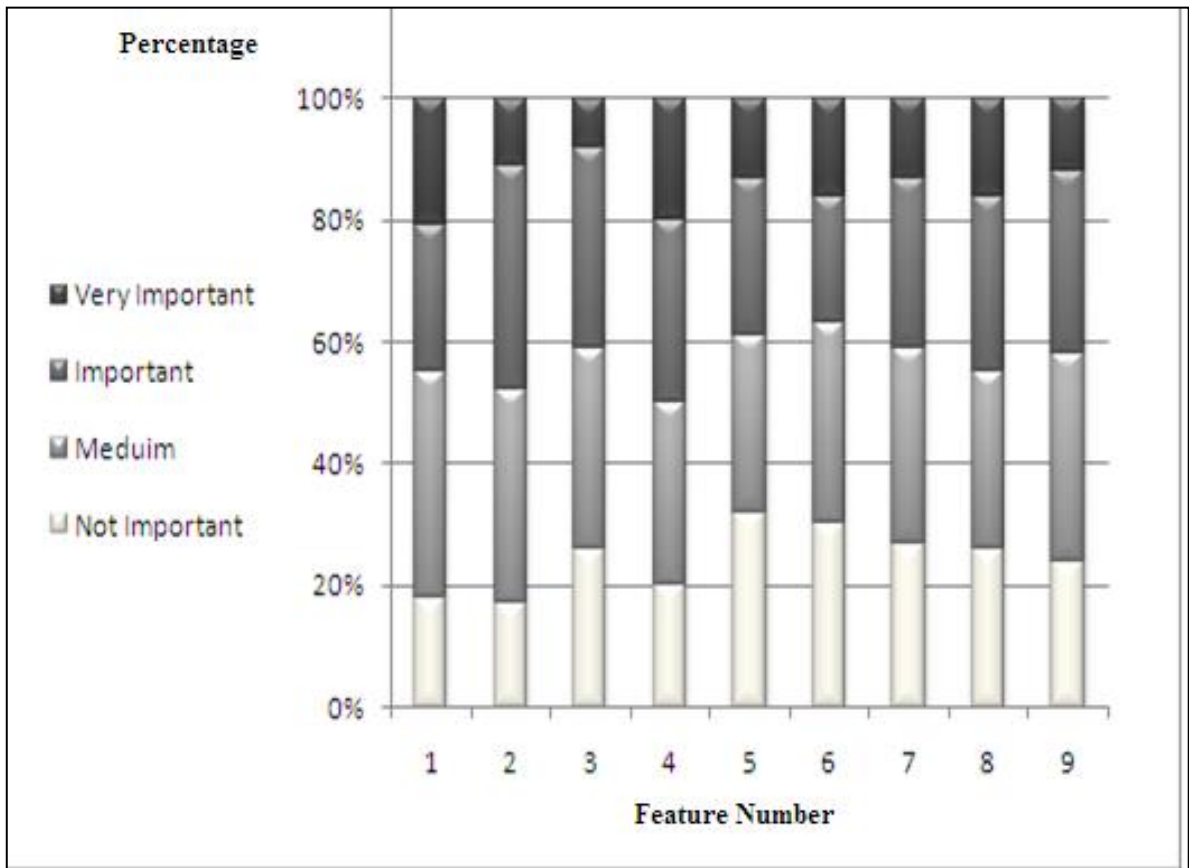


Figure 5: Rating of the Suggested Genealogical Features

Table 4 shows the percentage score for each genealogical feature. For each genealogical feature, there are four choices. The percentage for each choice is computed and the final score for each genealogical feature is computed according to the following equation:

$$Score = \frac{Sum}{4} \times 100\% \dots \dots \dots 1$$

Where

$$Sum = \sum_{i=1}^4 Percentage\ i \times (5 - i)$$

Percentage 1 = very important percentage

Percentage 2 = important percentage

Percentage 3 = medium percentage

Percentage 4 = not important percentage

Table 4: Questionnaire Results

| No. | Feature | Very important | Important | Medium | Not important | Score |
|-----|-----------------------------------|----------------|-----------|--------|---------------|-------|
| 1 | Family tree | 21% | 24% | 37% | 18% | 62% |
| 2 | Knowledge the family roots | 11% | 37% | 35% | 17% | 61% |
| 3 | Accessing the family members | 8% | 33% | 33% | 26% | 56% |
| 4 | Separation of Friend and relative | 20% | 30% | 30% | 20% | 63% |
| 5 | Relative notification | 13% | 26% | 29% | 32% | 55% |
| 6 | The Relative Inbox | 16% | 21% | 33% | 30% | 56% |
| 7 | Families gathering | 13% | 28% | 32% | 27% | 57% |
| 8 | Relative verification | 16% | 29% | 29% | 26% | 59% |
| 9 | Collaborative family tree | 12% | 30% | 34% | 24% | 58% |

3.5 Requirements

System requirements are often classified as functional requirements and non-functional requirements. Functional requirements are selected based on the genealogical features that result from the questionnaire, and at the end, we introduce the use-case to illustrate the adopted functional requirements.

3.5.1 Functional Requirements

According to people responses about genealogical features, the functional requirements are ranked and prioritized based on their scores as follows:

- **Separation of friend and relative:** Once the user joins family group, he will invite his relatives to join the group. His relatives will be members of the family group and they will be identified indirectly as his relatives.
- **Family tree:** Each family group has family tree where family members can edit, update its information, or build their family group.
- **Knowledge of the family roots:** Every family group member will be able to access the family tree and know the family roots, this is will be implemented indirectly in family view
- **Relative verification:** Family group administrator should verify the user accounts before adding them to the family group, this is will be implemented through administrator privileges.
- **Collaborative family tree:** Each family group has a wall where family members can share activities and events.
- **Families gathering:** This can be implemented by grouping the family groups that have similar roots in united family group. That requires large database and relationships and this will be left for future work.
- **Accessing family members:** Every family member can access family tree, also can access family members' accounts indirectly by clicking on their names on family tree view.
- **Relative notifications:** This can be implemented by rearranging the user notifications based on relations. The first-degree relative's notification, and then the second-degree and so on. This will be left for future work.

Table 5 lists the genealogical features, their rank, their score, and whether they will be implemented, implemented with family tree indirectly or not implemented.

Table 5: Feature Rank and Implementation Decision

| Rank | Score | Feature | Decision |
|------|-------|-----------------------------------|--------------------------------|
| 1 | 62% | Separation of friend and relative | Will be implemented indirectly |
| 2 | 61% | Family tree | Will be implemented |
| 3 | 56% | Knowledge of the family roots | Will be implemented indirectly |
| 4 | 63% | Relative verification | Will be implemented |
| 5 | 55% | Collaborative family tree | Will be implemented indirectly |
| 6 | 56% | Families gathering | For future work |
| 7 | 57% | The relatives inbox | For future work |
| 8 | 59% | Accessing family members | Will be implemented indirectly |
| 9 | 58% | Relative notification | For future work |

3.5.2 Non-Functional Requirement

The application is supposed to accomplish the following non-functional requirement:

- **Scalability:** The application will have ability to have a large number of members in the same tree that means the tree can be expanded.
- **Usability:** The application will be user friendly through the graphical user interfaces that direct any beginner user to create his family tree.
- **Open:** The application will be implemented as a web based application which means that it is open and available for any user.
- **Performance:** The response time will be compatible with user needs that may be assured because of features of programming language and the database that will be used to implement the application.
- **Reliability:** The application will process efficiently with less errors and corruptions.

3.6 Use-Case Analysis

Use-cases are scenario-based technique for requirements elicitation. They identify the type of usages and actors in a simple way. Figure 6 illustrates the use-case diagram of the suggested system. In Figure 6, there are eight use-cases and three actors. Our approach to fulfill the needed requirements is through a feature called Family Group (FG) that will be described further below.

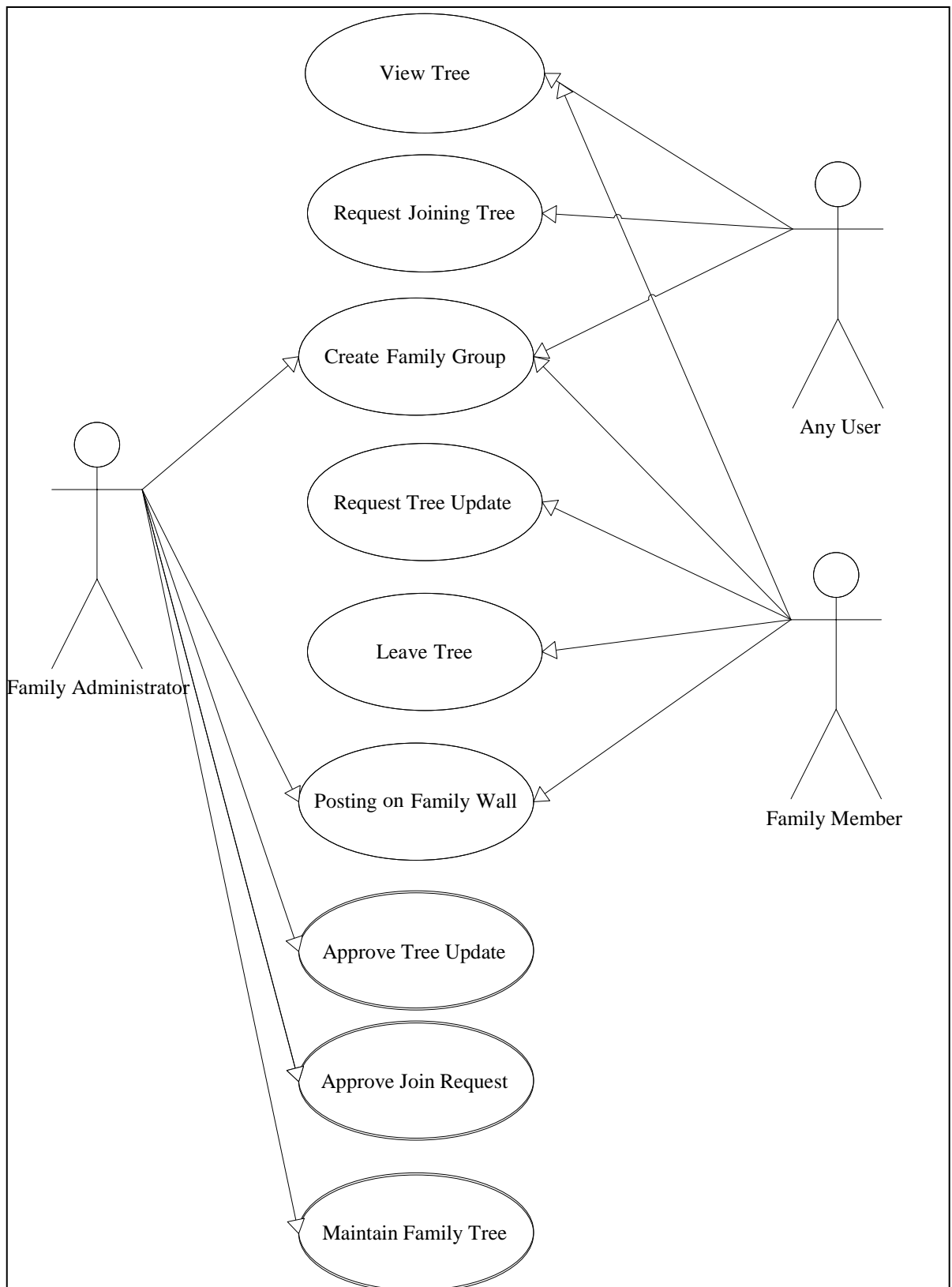


Figure 6: Use-Case Diagram of FG

3.6.1 Use-Case Actors

The FG has the following actors:

- **Any user:** The user who has a user account on SN and is not a member of the FG.
- **Family member:** The user who has a user account on SN and also is a member of the FG.
- **Family administrator:** The administrator of the FG is not only just a family member, he is also the creator of FG, or he is assigned as an administrator by the creator of the group.

3.6.2 Use-Case Scenarios

The FG is able to accomplish the following use-cases:

- **View tree:** The application should allow any user to know the family roots by viewing the tree in high resolution efficiently. The view will be clear enough even when the tree has a huge number of members. The user also can access other family members, even by clicking on their names in the hierarchy of the tree.
- **Request joining tree:** Any user can be a member of the FG. He can request from the administrator of the tree to join it. The user should identify his position in the family by choosing one person in already existent in the tree and the type of relation with this member. After the user joins the tree, he should be a family member and he should be able to see the family activities. He will have the family member privileges and use-cases.
- **Request tree update:** The application allows family members and the family administrator to add their relatives, delete them, or edit their information. The

member can also add other relatives who do not have user accounts. The relative might be his son, his wife, or his father. He can add them to the FG by filling the following data:

- First name
- Last name
- Type of relation
- Gender

In case of a family member, the data will not be on the tree. Instead, it will be kept in a temporary storage until it is approved (published by the administrator of the FG).

- **Post on family wall:** The FG has a wall where the family members can share activities and information, and collaborates on family events and activities. When a family member posts anything on the wall, the post will be viewed by all family members.
- **Leave tree:** Any family member can leave the family tree, without approval from the administrator. When the family member leaves the FG, he or she becomes a normal user, without family user privileges.
- **Approve tree update:** The application should allow the administrator of the FG to examine any recent modifications of tree and approve or delete these modifications.
- **Approve join request:** The application allows the FG administrator to approve any user request to join the tree; the administrator should verify the user profile by clicking on his profile and checking his information and his relation to the family.

- **Create FG:** Every user on the SN can create a FG, build its family tree, and add other family members to the FG.
- **Maintain family tree:** The application allows the administrator to update the family tree. The update will be viewed directly and will not be kept in the temporary storage.

Chapter 4. FG Design

4.1 Introduction

Depending on FG use-case, the GUI was deigned, the tree algorithm was chosen and implemented, and the database was designed and built. In this chapter, we will discuss FG GUI, database and used tree algorithm.

4.2 GUI Design

The Facebook *group* was chosen to be investigated with some genealogical features, the user should be login to use FG. Depending on our use-case, the GUI for FG was designed. All screens are designed with the bootstrap technology and HTML tools.

4.2.1 Any User

Figure 7 shows the FG view for any user.

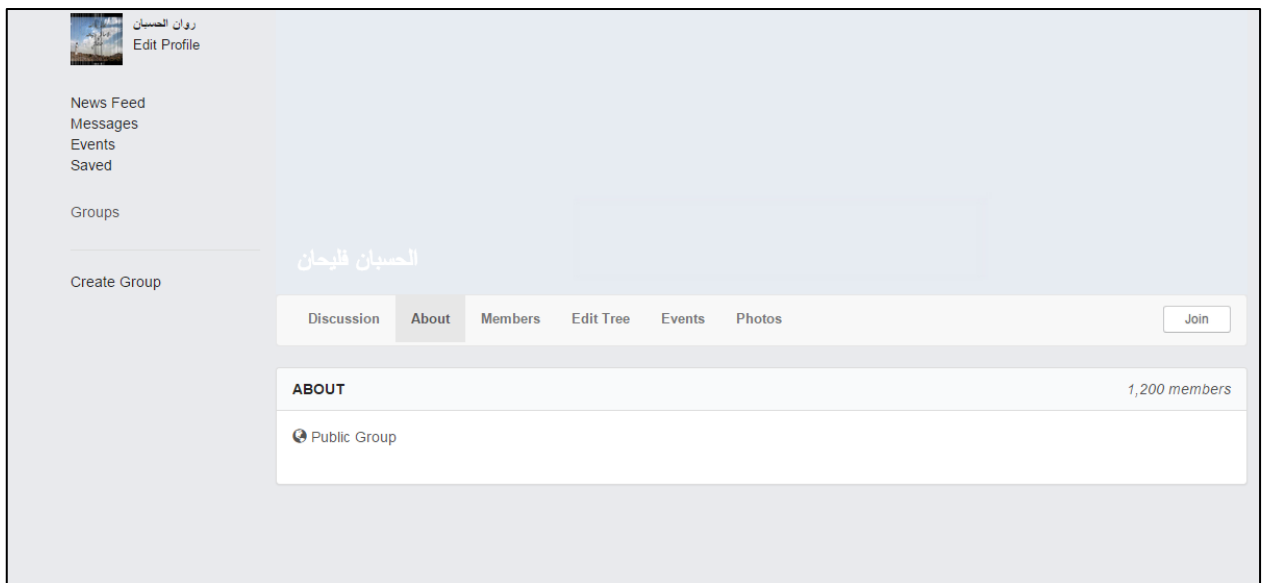


Figure 7: Any User Screen

After the user view the FG, he can join the family by clicking on join family button, and then the family tree will appear to pick his position and relation in the family tree as shown in Figure 8.

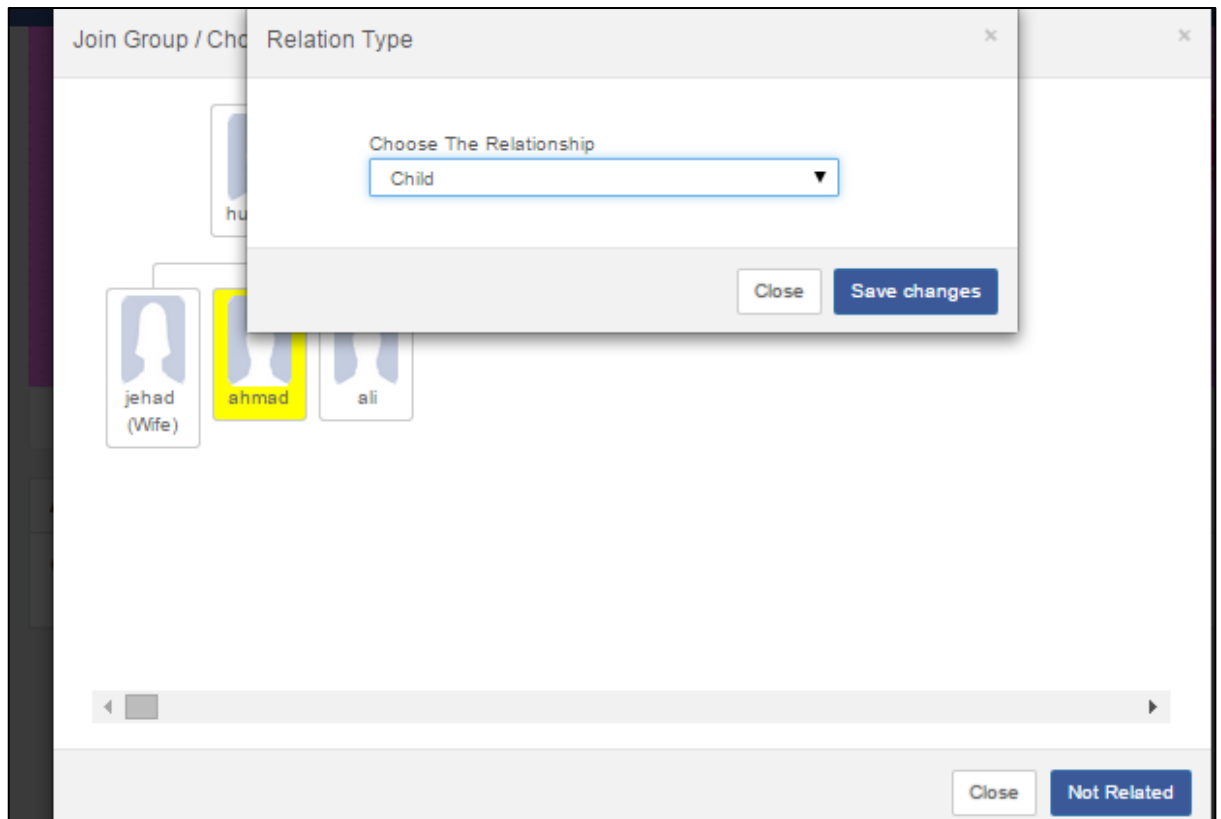


Figure 8: Join FG Screen

Any user can also create a FG by clicking on the Create FG links shown in Figure 9 and fill the following information:

- The group name
- The root information :
 - First name
 - Second name
 - Last name
 - Gender

Figure 9: Create FG Screen

4.2.2 FG Member

After the administrator approves any user request to join the group, the user will be added to the tree and the group view will appear as shown in Figure 10.

Figure 10: Family Member Screen

As shown in Figure 11, the family member can post on the FG wall. Also the family member can request adding or deleting nodes on the family tree by picking the node and entering the information shown in Figure 12.

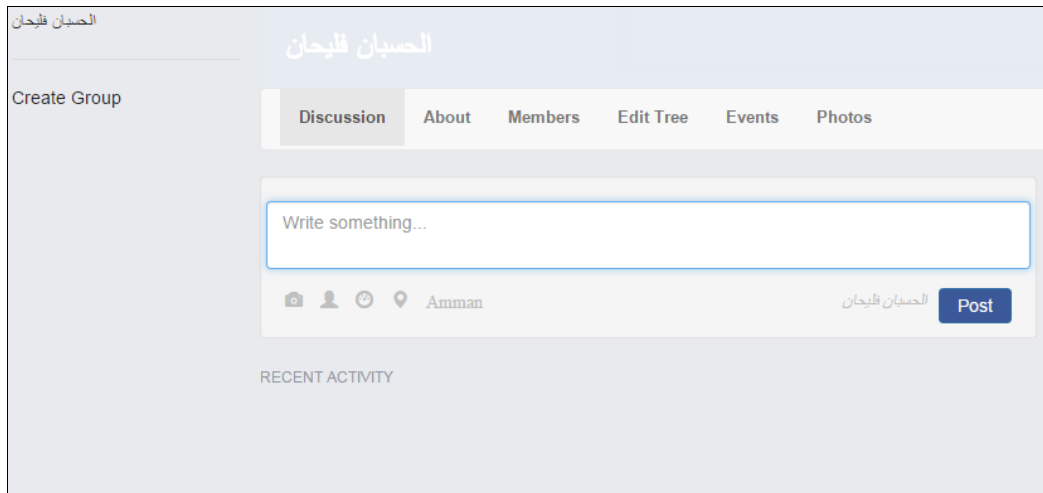


Figure 11: FG Wall Screen

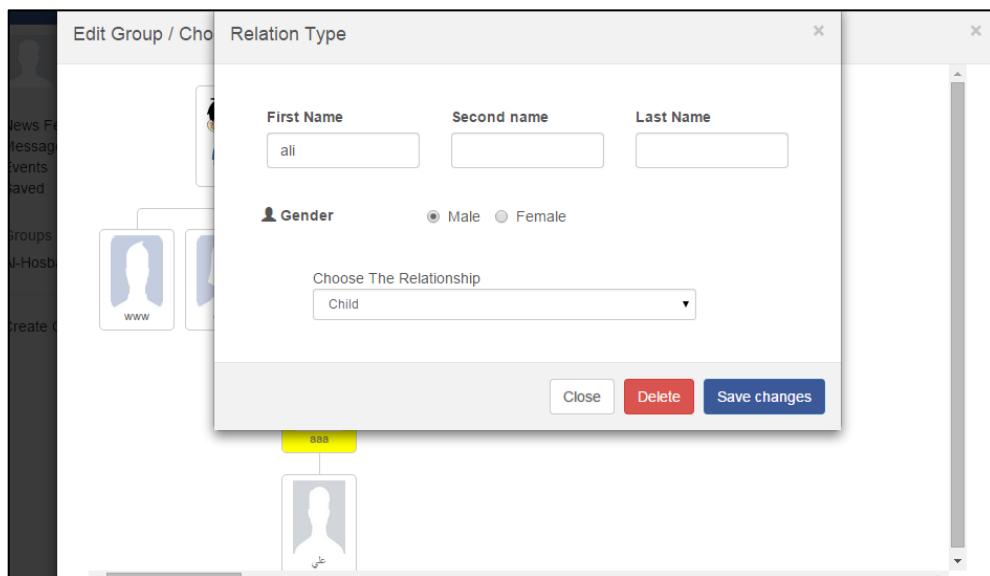


Figure 12: Request Family Update Screen

The family member can also leave the FG by clicking on the leave group button, once family member leaves the tree, he will not be accessed by other family members, but the user information and will be kept on the tree.

4.2.3 FG Administrator

The administrator receives join requests for the FG as shown in Figure 13. The administrator can directly approve or deny the request or view the join request and checking the user information by accessing the user profile.

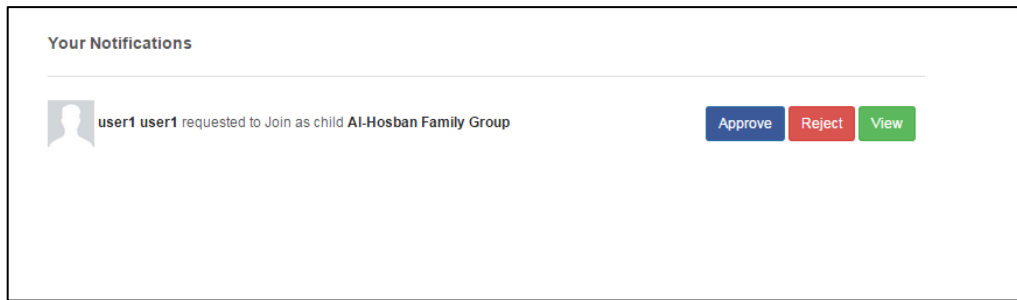


Figure 13: Join Requests Screen

By clicking on view button the Figure 14 appears.

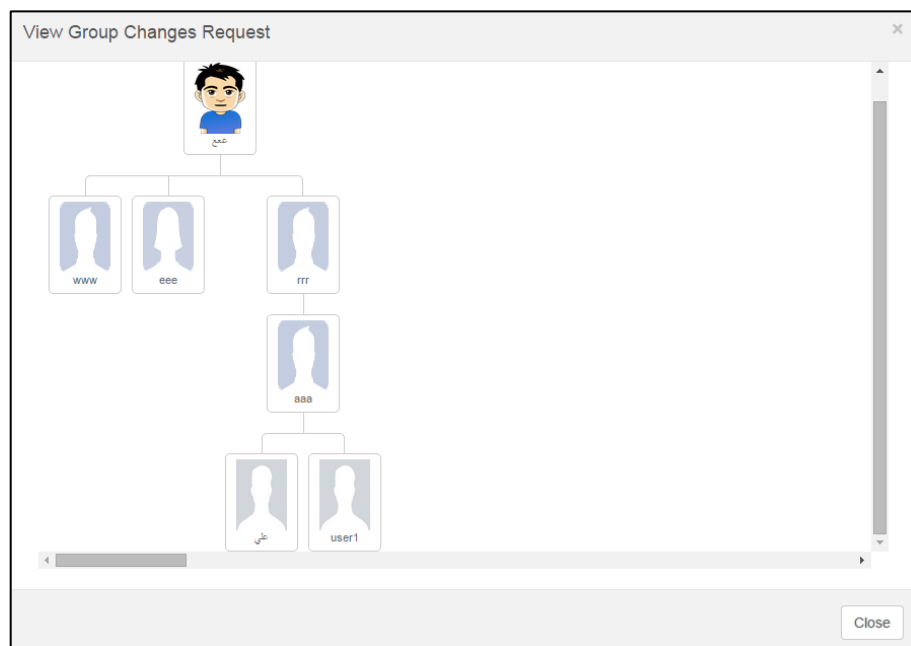


Figure 14: View Join Request Screen

The administrator can approve and reject tree updates as shown in Figure 15.

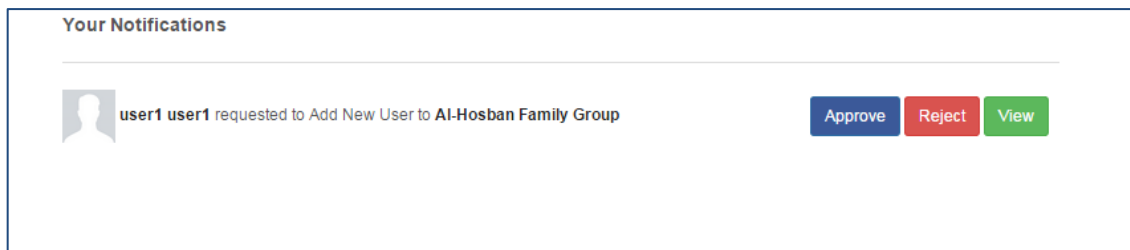


Figure 15: Tree Update Screen

The administrator can see the update by clicking on the view button, Then the administrator can decide to accept this update or not as shown in Figure 16.

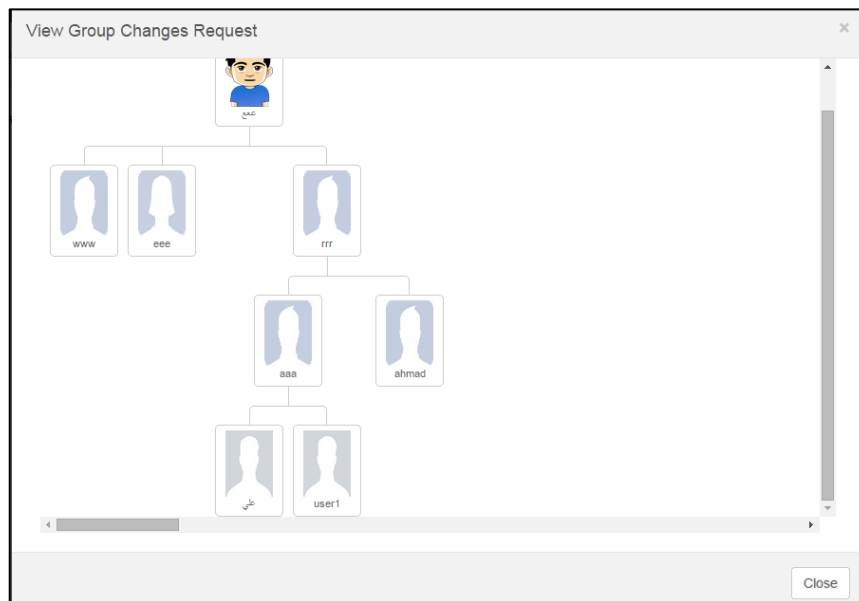


Figure 16: Approve Request Update Screen

The administrator can also maintain the family tree by adding, deleting, and editing relative information on the tree. Once the administrator clicks on any member of the tree, the following dialog box appears as shown in Figure 17. He can add them to the FG by filling the following data:

- First name
- Last name
- Type of relation
- Gender

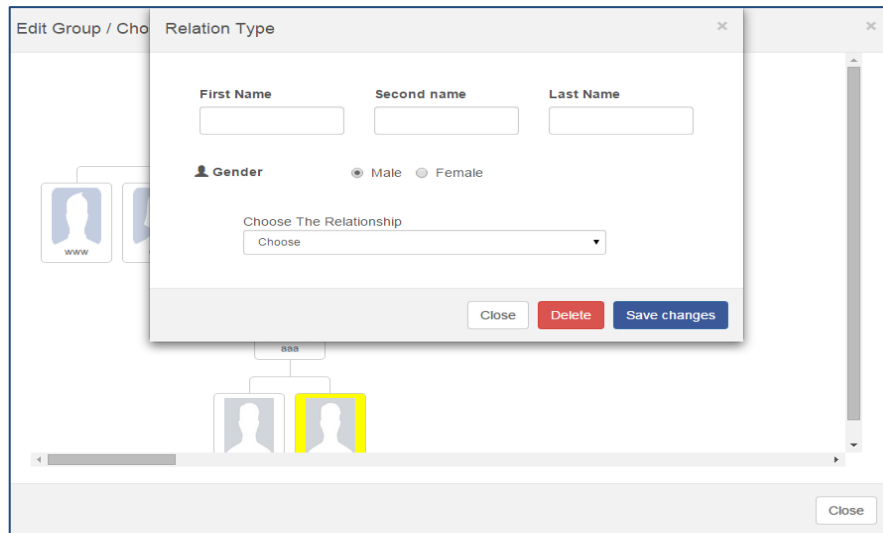


Figure 17: Maintain Family Tree Screen

The administrator can also assign other administrators by clicking on the manage tap and assign administrators from the family members list as shown in the following figure.

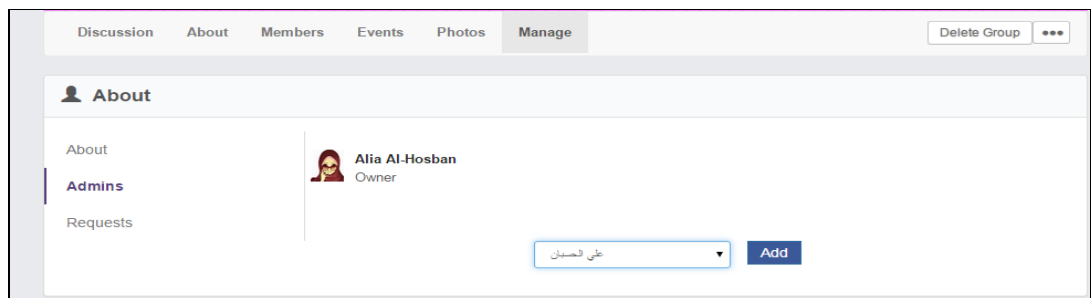


Figure 18: Manage Tap Screen

4.3 Family Tree Algorithm and Implementation

4.3.1 Introduction

The family tree used in FG is non-binary tree, we choose the DFS-traversal algorithm to draw a family tree for reasons discussed in Chapter 2. For each node in the tree, the following attributes are used: user ID, children IDs and parent ID. This link family node with each other.

4.3.2 Family Tree Algorithm

The algorithm gets the group members information from the user and member table. It stores them into an array, it also gets the tree root and its children, then it draws the root in the first time, and for each child of the current node, the following procedure will be taken:

If the current node has children, the draw function will draw one child of the current node, gets its information and gets its children, and calls itself. The algorithm is discussed in Figure 19.

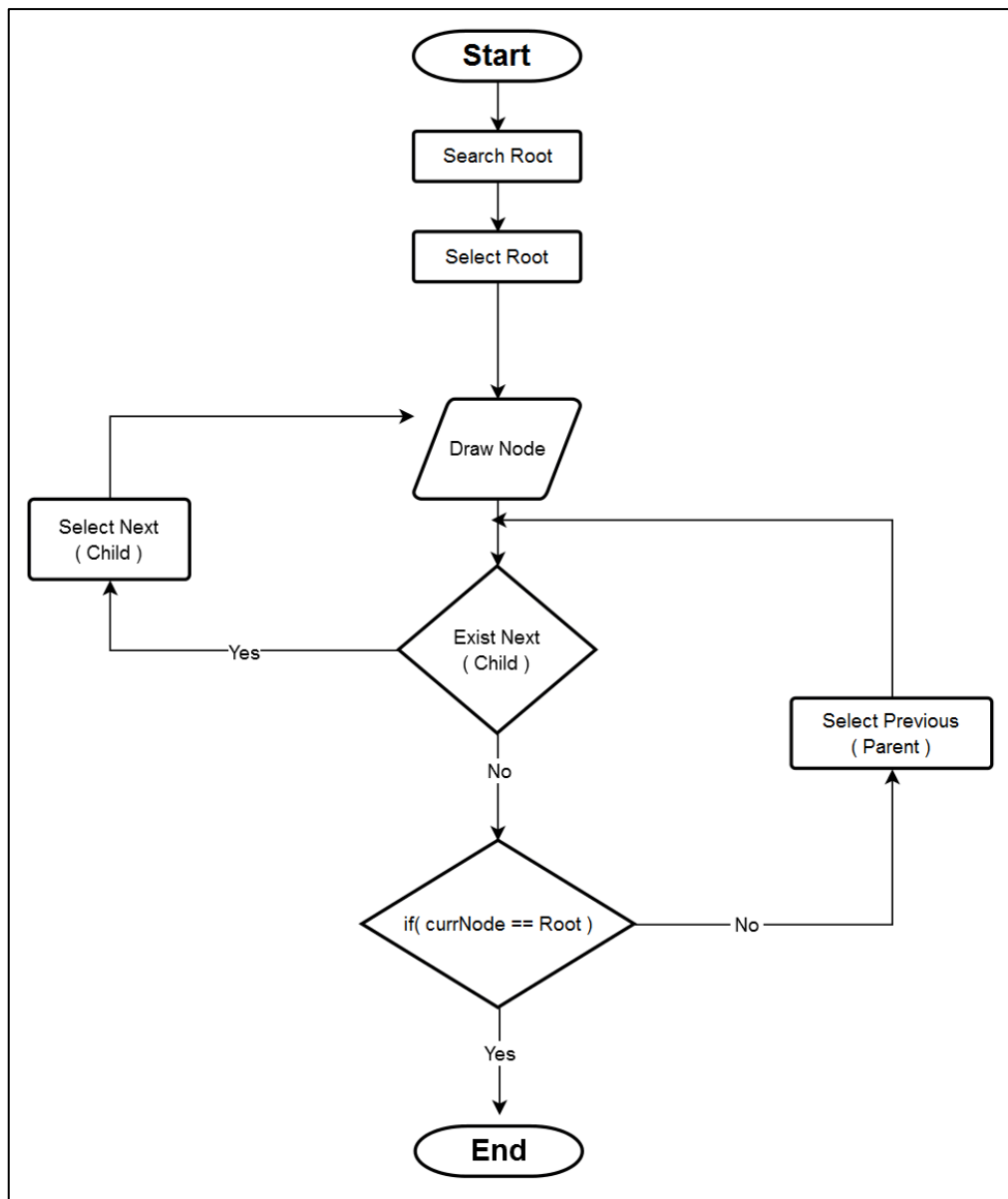


Figure 19: Family Tree Algorithm

4.3.3 Family Tree Implementation

Referring to the previous algorithm, the family tree split into five algorithms: The first algorithm saves the family tree members information which belongs to the same group in an array.

```

Function get_tree_members ($groupID) {
    $treeArr=array();

    $sel_query="SELECT    users.id
id,name,f_name,parent_id,children_ids,photo_url,is_user,is_member,gender
                        FROM  users,groups,member
                        WHERE users.id=relation_type.user_id
                        AND    groups.id=$groupID";

    $res=mysql_query($sel_query);
    while($row=mysql_fetch_assoc($res)){
        $treeArr[$row['id']]['name']=$row['name'];
        $treeArr[$row['id']]['f_name']=$row['f_name'];
        $treeArr[$row['id']]['parent_id']=$row['parent_id'];
        $treeArr[$row['id']]['children_ids']=$row['children_ids'];
        $treeArr[$row['id']]['photo_url']=$row['photo_url'];
        $treeArr[$row['id']]['is_user']=$row['is_user'];
        $treeArr[$row['id']]['is_member']=$row['is_member'];
        $treeArr[$row['id']]['gender']=$row['gender'];
    }
    return $treeArr;
}

```

The second algorithm finds the root of the tree.

```

{secondalgorithm:functionget_root($treeArr){
    foreach($treeArr as $key=>$node_arr){
        if($node_arr['parent_id'] == 0) return $key;
    }
}

```

The third algorithm saves user children IDs array of integers:

```
function strToArr($str){
    $childred_arr=array();
    if($str!="")$childred_arr=array_map('trim', explode('|',trim($str,'|')));
    return $childred_arr;
}
```

The fourth algorithm draws the tree; it passes the root of the tree and the tree members in first time. The draw function follows the following procedure:

- Gets the root children from member table
- Stores them into an array
- Draws the first child node by getting its name and its gender
- Puts them on the tree
- Defines different classes based on node gender. Tree class is defined for nodes and lines in the tree
- For each child of the current node, the following procedure will be taken: If the current node has children, the function will get its children, draw one of them, and calls itself.

```
functiondrawNodes($treeArr,$node_id){
    $node_arr=$treeArr[$node_id];
    $name=$node_arr['name'];
    $style="style='width:50px;height:70px'";
    $class=($node_arr['gender']?"class='Male'":"class='Female'";
    $popUp="role='button' data-toggle='modal'";
    $path="assets/img/profile/";
```

```

if($node_arr['gender'])$pic = 'M.gif';
else $pic = 'F.gif';

if($node_arr['is_user'] &&&& !empty(trim($node_arr['photo_url'])))
    $pic = $node_arr['photo_url'];
$src=$path.$pic;
?>
<li>
    <a href="#postModal" <?=$class?><?=$popUp?>><imgsrc="<?=$src?>" alt="No
Pic" <?=$style?>><br><?=$name?></a>
    <? $children_ids=strToArr($node_arr['children_ids']);//return array of Children IDs
    if(!empty($children_ids)){?>
        <ul><?
        foreach($children_ids as $node_id){
            drawNodes($treeArr,$node_id);
        }
    }

```

The fifth algorithm is *Drawtree*. It uses the draw nodes function by passing the group ID.

Drawtree ()

\$first node=get root (group id);

\$tree members= get tree members (group id);

Draw node(\$treemembers, \$first node);

4.4 Database Design

The FG requires tables for drawing tree and also tables for the SN information. The suitable tables for this purpose are the following:

- **User:** Contains the main information about users which are:
 - **ID:** The identifier for each user on SN, it is primary key in user table and it is referred to user ID in post table, in comment table and in like table

- Email: The email of the user
- Password: The password for the user
- Name: Display name on the tree
- Lname: Last name for user
- Fname: First name for user
- Bdate: Birth day for the user
- Gender: The gender of the user
- Photo_url: The path location for the user photo
- Cover url: The path location for the cover photo
- Isuser: Flag to identify the user on SN if it objects created by other user through drawing family, or already has a user account on SN
- Phone number
- Home town
- Living place
- Education
- Workplace
- **Group:** Contains the main group attributes which are
 - Group ID: Group identifier which is the primary key in group table and foreign key in post table
 - Group name
 - Group information: Contains the main information about a group which is inserted by the owner
 - Owner: The group creator, he has the administrator privileges
- **Post:** Contains the FG posts, which has the following fields:
 - User ID: is a foreign key in post table

- **Group ID:** Is the foreign key in post table
- **Post ID:** primacy key in post table
- **Post text:** The contents of the post text
- **Post-date:** The date of the post
- **Like:** Contains the member likes for different group posts, it has the following fields:
 - User ID
 - Group ID
 - Post ID
- **Comment:** Contains the user's comment for group posts, it has the following fields:
 - Comment ID
 - Group ID
 - Post ID
 - Comment text
 - Comment date
- **Member:** Contains the group members and their relations with other members, it has the following fields:
 - Group ID
 - User ID
 - Parent ID: This will save the parent IDs to use them in drawing the tree recursively.
 - Children ID: This will save the children IDs for every user ID, this field will help us to draw the family members in DFS/Perorder algorithm

- Isadmin: Flag has three values 0 not administrator, 1 administrator, 2 owner
- Wifeid: The ID for wife
- **Notification:** contains the main notifications that are related to FG, it has the following fields:
 - ID: Notification ID
 - Requester ID: the ID for the requester, it is the ID for active user or has a session with FG
 - Requested ID: The ID of requested added node or -1 if the request type is delete
 - Request type: Join, edit, delete, or add
 - Relation type: Self, wife, children, root, or not related.
- **Pending request:** is copy of member table, in case of a family member, the tree Updates will be saved in Pending table, it will saved in member table after administrator approval. The table contains the following fields:
 - Notification ID
 - Group ID
 - User ref: Is as a user ID in the member table=requester ID =session ID
 - Parent ID
 - WifeID
 - Is admin

Figure 20 shows the database design. After many tries to write the algorithm, and after searching in depth, I designed the database which summarizes lots of additional

information, and also links between the family tree and SN at the same time, which is the main issue in our thesis.

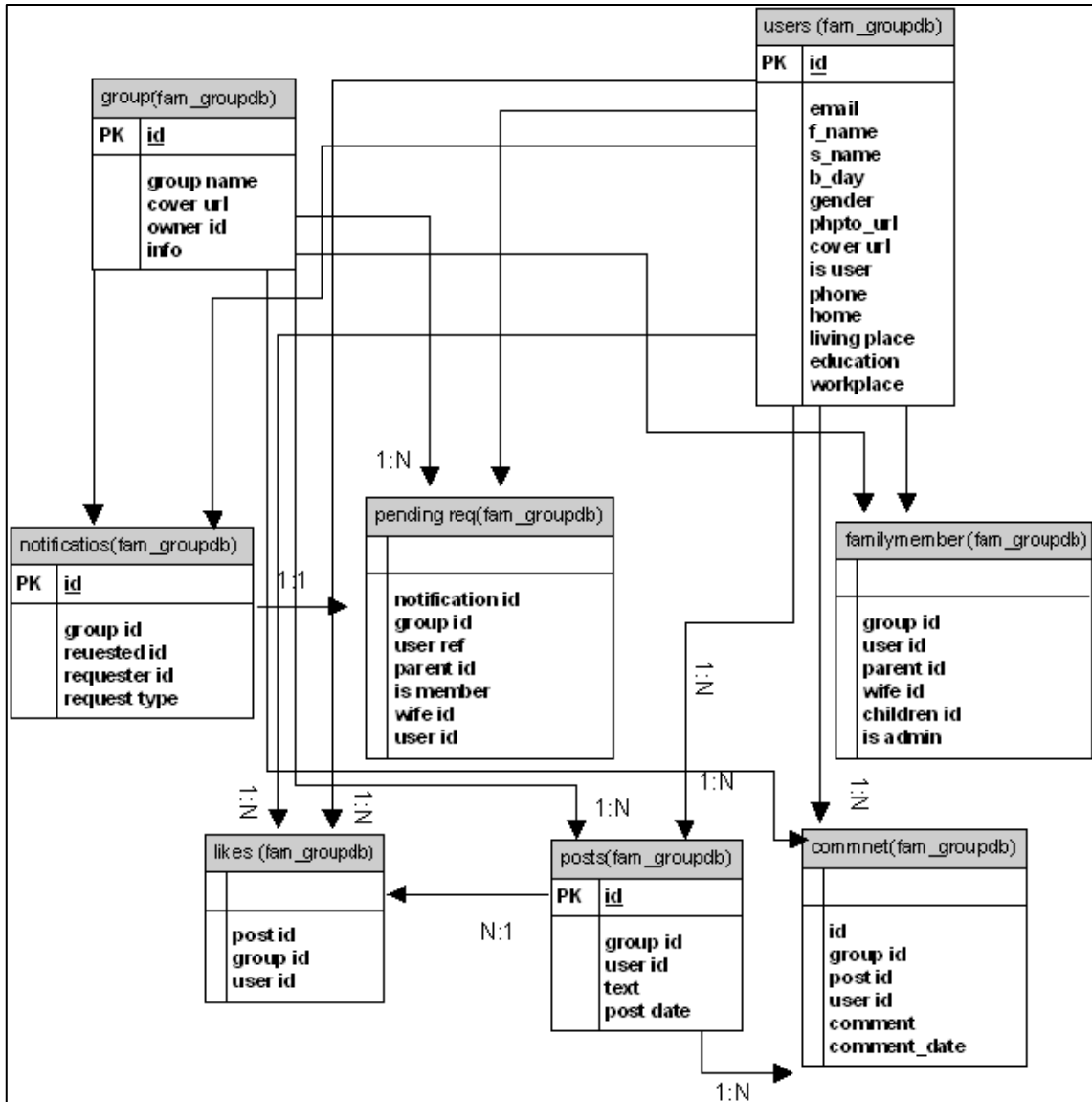


Figure 20: Database Design

Chapter 5. GUI Programming and Tree View Programming

This work consists of two parts GUI "form" programming and the tree view programming.

5.1 The GUI Programming

5.1.1 Introduction

The graphical user interface plays an important role in helping users to interact with the FG. We choose the Facebook as SN to investigate with suggested genealogical features that have been discussed in Chapter 3.

The first page that appears for any user is a Facebook welcoming screen. If the user already registered, he/she can login. Otherwise the user requests an account by filling registration form shown in Figure 21.

Figure 21: Sign Up and Login Windows

The welcoming page is implemented using four functions: *sign up*, *authenticate*, *createuser*, and *validateemail*.

The *signup* function takes the user information after he has been filled, it validates the uniqueness of the email using the *validate* function. It also calls the *createuser* function to insert the user information into the user table.

If the user wants to login in by using his email and password, the *authenticate* function will search for the user email in the users table and check the entered password.

After the user creates his account, Figure 22 appears. The screen shows his/her basic information. The user can edit his basic information and change his profile and cover page photos.

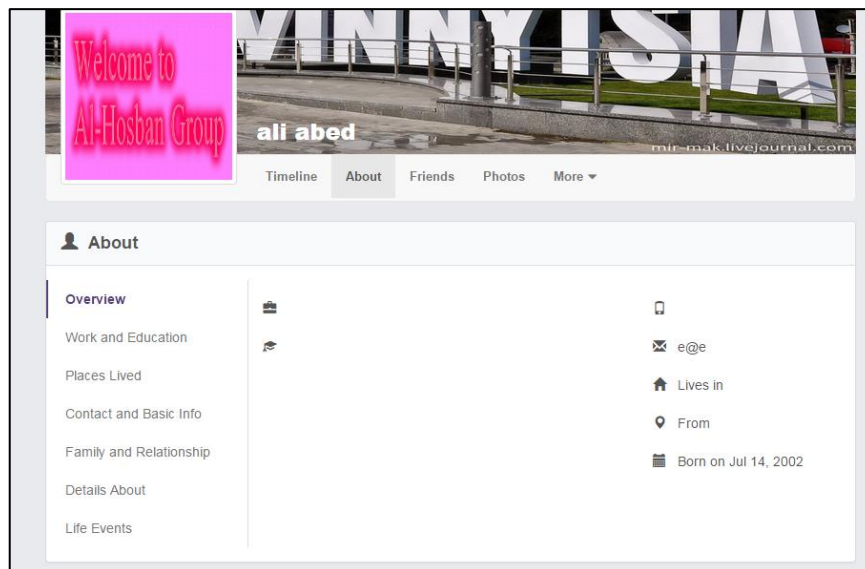


Figure 22: User Profile Window

Also the user can edit his information, and the updated information will be inserted into users table as shown in Figure 23

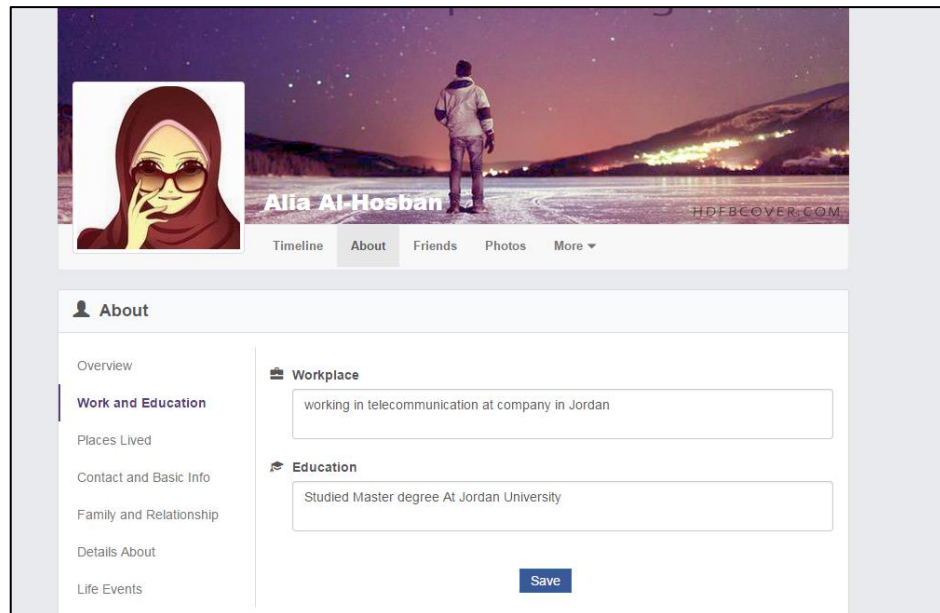


Figure 23: Edit Profile Information

The user can also search for available groups and profiles by typing their names as shown in Figure 24, the groups will be ordered alphabetically

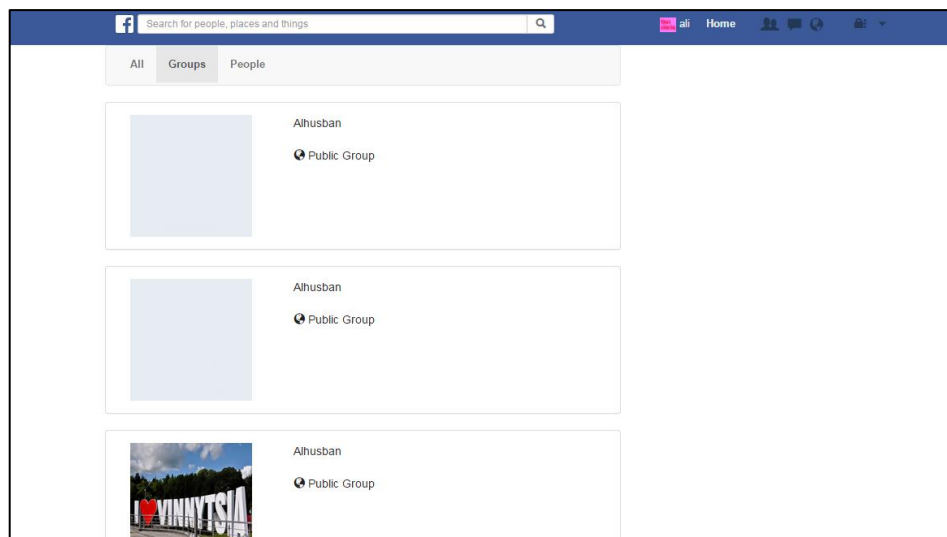


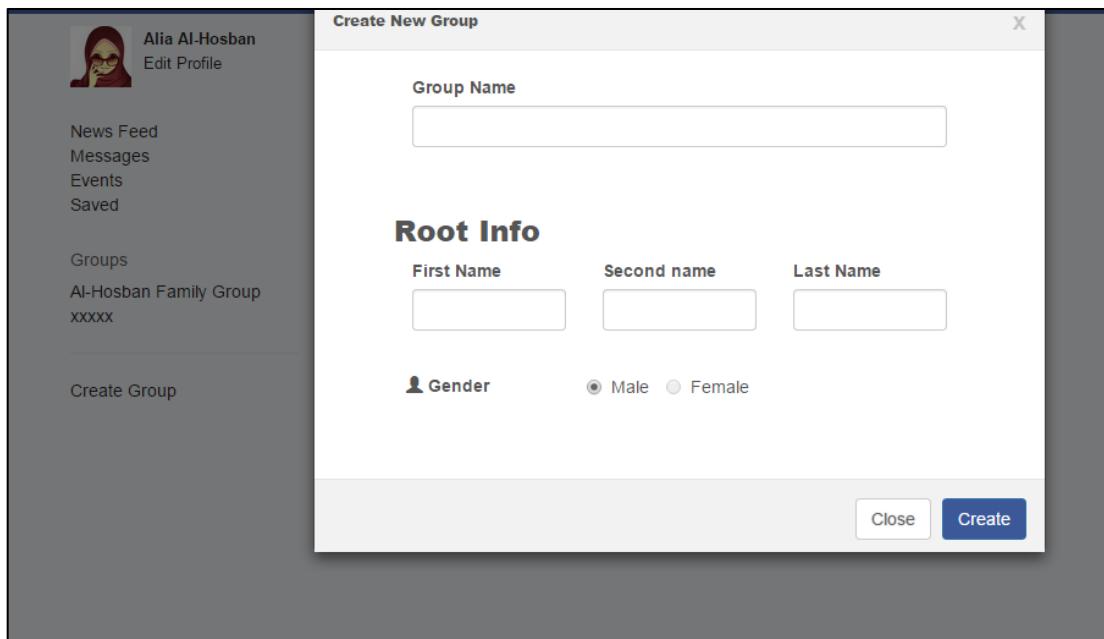
Figure 24: Search Group Window

The searching operation uses the *search* function. It takes the keyword typed as input and returns an array of available groups or profiles or both.

The *search* function has the following types:

- Type 0: Search the keyword in group table and user table
- Type 1: Search the keyword in user table
- Type 2: Search the keyword in the group table
- If no one exists, the result will be "no values exist"

Also the user can create groups as shown in Figure 25. It is implemented using *creategroup* function, which assigns the ID for the group using *maxId* function as well as It returns the maximum ID from the group table, and assigns the maximum ID+1 for created group ID.



The screenshot shows a user interface for creating a new group. The user is Alia Al-Hosban. The dialog box contains the following fields and options:

- Group Name:** A text input field.
- Root Info:**
 - First Name:** A text input field.
 - Second name:** A text input field.
 - Last Name:** A text input field.
- Gender:** Radio buttons for Male (selected) and Female.

Buttons: Close, Create

Figure 25: Create Group Dialog Box

Once user creates new family group, the owner information, root information will be inserted into user table and member table as mentioned in Tables 6, 7, 8.

Table 6: Initial Values for Group Owner

| Field | Initial Value |
|-----------|-------------------------|
| Group ID | The max ID in group |
| User ID | Session ID |
| Parent ID | -1: not related to tree |
| Wife ID | -1: the default ID |
| Is admin | 2: the owner |

Table 7: Initial Root Information for Created Group

| Field | Initial Value |
|-------------|----------------------------------|
| ID | The max ID from the user table+1 |
| First name | First name for the root |
| Second name | Second name for the root |
| Last name | Last name for the root |
| Isuser | 0, the root is object not user |
| Gender | The root is male |

Table 8: Root Information

| Field | Initial Value |
|-----------|-----------------------------|
| User ID | The max ID for user table+1 |
| Parent ID | 0, because it is the root |
| Is admin | 0, not administrator |

5.1.2 Group Structure

As well as group contains many sections to cover all use-case scenarios and actors, the following sections are assigned to each actor:

- Any user: About, member, and join group button
- Family member: About, discussion, edit member, member and leave group button
- Administrator: About, discussion, edit member, member, manage, and delete group button

The group views for different users are discussed in Figures 26, 27 and 28. The group header gets the group information from member table and assigns the following variables to distinguish among different kinds of users:

- Is member: 1, for all group members that belong to given group and store them into an array
- Is admin: Flag has one of the following values:
 - 1: Administrator
 - 2: Owner, the creator of FG and he can assign other administrators
 - 0: Not administrator or owner



Figure 26: Any User Group View



Figure 27: Family Member Group View



Figure 28: Administrator Group View

5.1.3 Wall

Many functions are programmed to make a collaborative wall between family members.

Wall functions and their description are mentioned in Table 9.

Table 9: Wall Functions and Descriptions

| Function | Description |
|--------------------|---|
| Post | Insert post into posts table |
| Comments | Insert comments into comment table |
| Delete comments | Delete comments from comment table |
| Delete posts | Delete posts and its related comments |
| Like | Insert records into likes table |
| Unlike | Delete one record from like table |
| Get group posts | Get all group posts from the post table based on group ID |
| Get group comments | Get all group comments based on post ID and group ID |

Besides the family member can comment and like a post, he can delete his own post. On the other hand the group owner can delete group posts and comments.

5.1.4 Manage

Manage section contains the following parts: Assign other administrators, manage requests, and update group information, and the administrator can manage user requests based on their types, which contains the following:

- Request type =1, request to join
- Request type=2, request to add new user
- Request type=3, request to edit the object
- Request type=4, request to delete user from

The administrator can approve requests as well as he can view them. *Approve reject* and *view* functions and their descriptions are mentioned in table 10:

Table 10: Requests Functions and Descriptions

| Function | Description |
|------------------|--|
| Approve requests | Based on request type one record will be inserted into member table |
| Reject requests | Where one notification record will be deleted from the pending table and |
| View requests | Where tree will be viewed and updated. |

5.2 The Tree View Programming

The tree programming enters two phases. The first one to outline the desired interaction that is meant to achieve, and the second phase is the implementation phase.

5.2.1 User Interaction

What was planned to achieve in this part, is to enable the family member to easily navigate the family tree simply by clicking on that node and accessing its profile. He will have the ability to interact directly with the tree through clicking on that node with option: delete node, add child/wife, ability to display different genders with different

node colors, the node size is dynamically adjusted to fit the node name, and ability to view the node titles in Arabic language.

5.2.2 Implementation

The family tree is implemented in two ways: the first is view tree and the other is edit tree. The view type gives the members' ability to access the other family members' profiles. The other type gives the user ability to edit and select nodes, once user click on one of the family members on a tree it will be colored, we will discuss them in the following sections.

Once the user views the family tree, the family root will be got and the other group members will be also got by using *joinedgroup* function. Also family tree can be updated by insertion or deletion. The *Add/Remove* function will update the tree by changing the user children according to request.

- **View Tree**

Any user, family member and administrator can view family member group by clicking on member tab, once user clicks on any node in the family tree, he will be directed to node profile. The view tree section is implemented by calling *draw tree* function type 0.

- **Edit Tree**

Once the administrator or family member picks any node, the dialog box appears. Based on the kind of node, the following options are presented:

- User: Add child, add wife, delete user.
- Object: Add child, edit object, delete object
- Root (user): Add child, add root or father, add wife
- Root (object): Add child, add root or father, add wife, edit object

- Female (user): Delete user.
- Female (object): Add self, edit object.

If a node is selected to be deleted, all its descents will be deleted as well as the request will be inserted into a pending request table. In case of administrator approval, the edit request function will be called; it inserts a record in the notification table containing the following fields:

- Type of request
 - Relation type
 - Requester ID
 - Requested ID: Is clicked node wanted to delete, add or edit.
- **Request To Join**

Once any user request to join FG, the draw tree function type 2 will be called and the user will choose the kind of relationship and the node where he/ she want to be added.

Based on relationship, the following algorithm will be taken:

If kind of relationship= self

Delete the current node and add new one. Insert record in notification table and pending request table.

Else if kind of relationship=2

Create node and take its information from the session user ID and add it as child by clicking node and insert the record into the notification table and pending request table.

Else if kind of relationship =3

Add it as wife and insert record into notification table and pending request table.

Else if kind of relation =5

Change the parent ID of session ID to 0 and make the old root as child to new one and insert record into notification table and pending request table.

- **Other Functions**

The following functions are also implemented in FG:

- **Leave group:** Once the user leaves a group, he will be deleted from the group and will not be deleted from the tree, he/she will be changed to object.
- **Delete group:** Once the owner of the group deletes the group, the tree will be deleted and all objects created in the tree will also be deleted.

5.3 Functional Requirements Implementation

We have discussed the GUI programming and family tree programming, the following genealogical features have been investigated in SN:

- **Separation of friend and relative:** Once the user joins family tree, he/she can see his/her relative tree discussed in tree view programming/edit tree.
- **Family tree:** Each FG has family tree where family members can edit, update its information, or build his own FG, it has been discussed also in family tree programming section
- **Knowledge the family roots:** Any user has ability to view the family tree root through viewing the tree.
- **Relative verification:** FG administrator should verify the user accounts before adding them.

- **Collaborative family tree:** Each FG has a wall where family members can share family events, which have been discussed in the GUI programming section.
- **Families gathering:** This can be implemented by grouping the family groups who have similar roots. That requires large database and relationships and this will be left for future work.
- **Accessing family members:** Every family member can access family tree, as well as he can access family member accounts by clicking on their names on family tree view, which have been discussed in Tree view programming type 1.
- **Relative notifications:** where user notifications were arranged based on relations. It can be implemented by using the information in the member table, and it will be left for future work.

Chapter 6. Requirements Validation and Evaluation

6.1 Introduction

After completing the site, the site is used by some volunteers to test the functional requirements. The volunteers were trained over using the site, and they gave enough time to use it, then we asked them to complete the questionnaire about FG (Appendix B).

The following sections present the evaluation of the developed site and volunteer responses about the functional requirements and non-functional requirements.

6.2 Functionality Validation

The questionnaire (Appendix B) contains nine use-cases. It investigates the people views about FG functionality. It also investigates the degree of satisfaction for each use-case in FG. The questionnaire was completed by 55 persons. The people answer questions about the satisfaction of each FG use cases by one of the following responses: Excellent, very good, good, poor. The questionnaire contains the following functional requirement:

- View tree
- Request joining tree
- Request tree update
- Post on family wall
- Leave tree
- Approve tree update
- Approve join request
- Create FG
- Maintain family tree

6.2.1 Data Collection (Appendix B)

The website was used by 55 persons (32 females and 23 males). The data collected is summarized according to the age and education level.

Figure 29 shows the participants distributions based on three categories: their sex, their age, and their education.

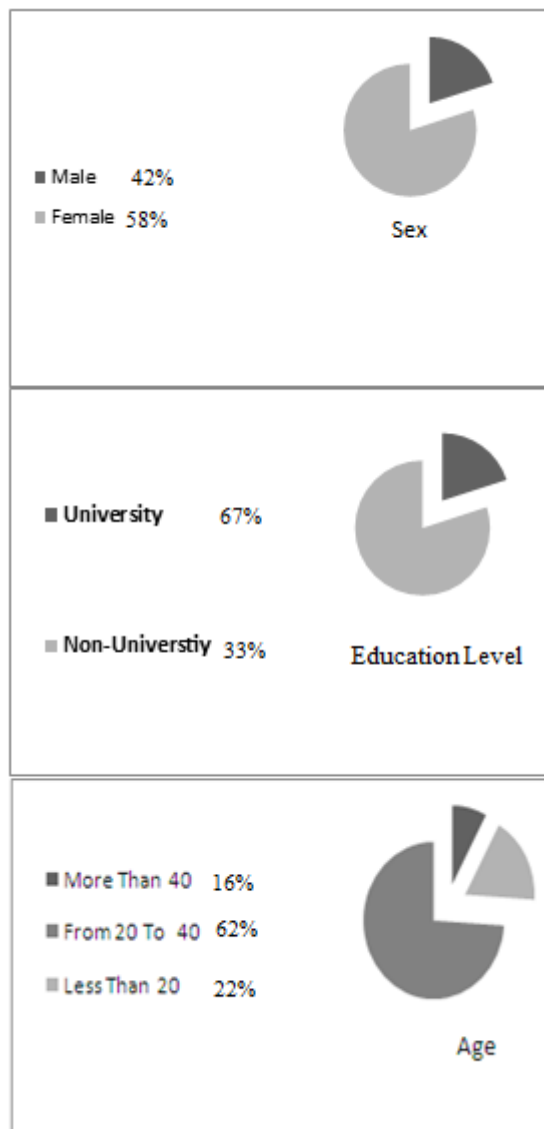


Figure 29: Distribution of Participants of FG Based on Sex, Educational Level, and Age

6.2.2 Results

According to the volunteer responses on the questionnaire (Appendix B), the approval percentage is shown in Table 11 and Figure 30. Table 11 shows the percentage score for each use-case. For each use-case there are four choices, the percentage for each choice is computed and the final score for each use-case is computed according to the following equation:

$$Score = \frac{Sum}{4} \times 100\% \dots \dots \dots 2$$

Where:

$$Sum = \sum_{i=1}^4 Percentage\ i \times (5 - i)$$

Percentage 1 = Excellent Percentage

Percentage 2 = Very Good Percentage

Percentage 3 = Good Percentage

Percentage 4 = Poor Percentage

Table 11: Questionnaire Results Percentages for Use-Cases

| No. | Use- Case | Excellent | Very Good | Good | Poor | Score |
|-----|----------------------|-----------|-----------|------|------|-------|
| 1 | View tree | 72% | 24% | 2% | 2% | 91% |
| 2 | Request joining tree | 72% | 18% | 5% | 5% | 89% |
| 3 | Request tree update | 60% | 33% | 5% | 2% | 87% |
| 4 | Post on family wall | 55% | 35% | 3% | 7% | 84% |
| 5 | Leave tree | 58% | 25% | 9% | 8% | 83% |
| 6 | Approve tree update | 60% | 20% | 15% | 5% | 85% |
| 7 | Approve join request | 56% | 27% | 6% | 11% | 78% |
| 8 | Create FG | 78% | 16% | 2% | 4% | 92% |
| 9 | Maintain family tree | 53% | 35% | 7% | 5% | 84% |

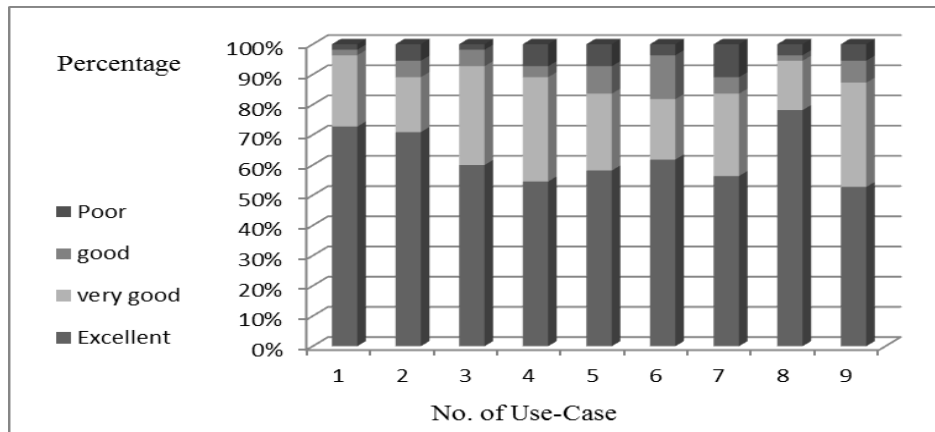


Figure 30: Rating of Evaluated FG Use-Cases

6.3 Non-Functional Requirements Validation

The questionnaire (Appendix B) contains five non-functional requirements of FG. It investigates the people views about FG requirements. It also investigates the degree of requirement satisfaction on FG. The questionnaire was completed by 55 persons. The people answer questions about the satisfaction of each non-functional requirement by one of the following responses: Excellent, very good, good, poor. The questionnaire contains the following non-functional requirement:

- Usability
- Open
- Performance
- Reliability

6.3.1 Results

According to the volunteer responses on the questionnaire (Appendix B), the approval results as percentages are shown in Table 12 and Figure 31. Table 12 shows the percentage score for each non-functional requirement, for each non-functional requirement there are four choices. The percentage for each choice is computed as in Section 6.2.2; all non-functional requirements are evaluated with a high score.

Table 12: Questionnaire Results Percentage for Non-Functional Requirements

| No. | None-Functional Requirement | Excellent | Very Good | Good | Poor | Score |
|-----|-----------------------------|-----------|-----------|------|------|-------|
| 1 | Usability | 87% | 9% | 2% | 2% | 95% |
| 2 | Open | 73% | 22% | 4% | 1% | 91% |
| 3 | Performance | 45% | 40% | 9% | 6% | 81% |
| 4 | Reliability | 38% | 44% | 11% | 7% | 78% |

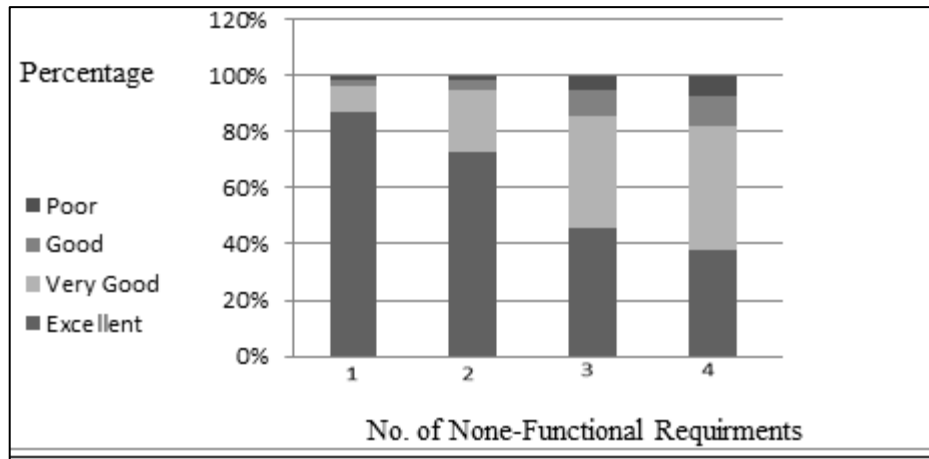


Figure 31: Rating of the Evaluated None-Functional Requirements

6.3.2 Scalability

The family tree algorithm has been implemented and run on a machine which has the following specification: 2.13 GHz Dual Core CPU, 2 GB Ram, the family tree algorithm is also tested for a large number of nodes; the execution time is measured for each test as shown in Figure 32 as a function of the tree size in number of nodes. All these tree sizes are drawn in a short period of time not exceed 0.2 seconds.

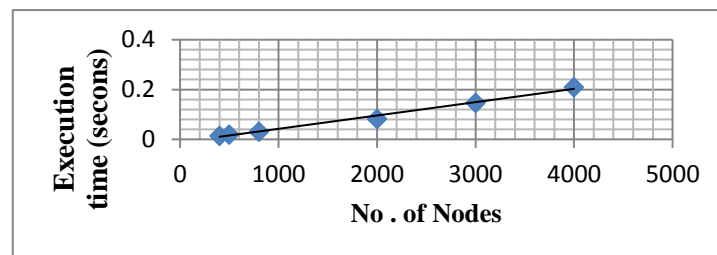


Figure 32: Execution Time for Different Tree Sizes

Chapter 7. Conclusions and Future Work

7.1 Conclusions

In this thesis, we studied the SN and genealogical sites. We started with a survey of the main services in SN and genealogical sites, and then designed an application in SN to provide some genealogical features. The application was designed to take the basic genealogical features into consideration. After the application design, we used a questionnaire about the genealogical features that people require in SN. The suggested genealogical features were ranked and implemented based on participant's responses. Some of the suggested requirements were implemented; others were left for future work.

We choose a Facebook group as SN to be investigated with genealogical features. FG is designed to accommodate the SN and the required genealogical features. FG was used by 55 volunteers to evaluate the basic requirements; all results were registered and scored. The results showed that FG functionality and its non-functional requirements have high score. Tree algorithm was also tested to evaluate system performance; it showed the ability to draw large number of nodes in a short period of time.

7.2 Future Work

The future work includes further investigation of the SN with more genealogical features. In case of FG the following genealogical features can be investigated:

- **Families gathering:** where this feature can be defined by grouping the family groups with similar roots in a united FG. The feature can be implemented by comparing the groups with parent ID = 0, if they have the similar information, they will be gathered in the same wall, and a tree is built to contain all the members, then the original groups will be deleted. This feature will require large database and relationships.

- **Relative notifications:** rearranging the user notifications based on relations. It can be implemented by using the following information from the member table: session ID, children ID, and parent ID.
- **Relative inbox:** adding an inbox to each FG, where the administrator can send private message to all family members.
- **Searching for family tree:** searching for one of the user relative relatives, will result with showing the relative profile, and the degree of relationship between the user and result profiles.

References

About.com Genealogy, Social Networking Sites for Genealogists. Retrieved April, 12, 2014 from http://genealogy.about.com/od/social_networking/tp/portals.htm.

Akanmu, T., Olabiyisi, S, Omidiora, E., Oyeleye, C., Mabayoje, M. & Babatunde, A. (2010). Comparative Study of Complexities of Breadth-First Search and Depth-First Search Algorithms Using Software Complexity measures. **In Proceedings of the World Congress on Engineering** (Vol. 1).

Barnett, G. &Tongo, L. (2008), **Data Structures and Algorithms: Annotated Reference with Examples**. Queensland University of Technology.

Cormen, T., Leiserson, C., Rivest, R., and Stein, C. (2001), **Introduction to Algorithms** (Vol. 2, pp. 531-549). Cambridge: MIT press.

Dwyer, C., Hiltz, S., & Passerini, K. (2007), **Trust and Privacy Concern within Social Networking Sites: A Comparison of Facebook and MySpace**. In AMCIS (p. 339), Korea

Ellison, N. (2007), Social network sites: Definition, history, and scholarship. **Journal of Computer Mediated Communication**, 13(1), 210-230. Retrieved May, 25, 2015 from Willy Online Library.

Ellison, N., Stein, C. &Lampe, C. (2007), The Benefits of Facebook “friends:” Social Capital and College Students’ Use of Online Social Network Sites. **Journal of Computer Mediated Communication**, 12(4); 1143-1168. Retrieved May, 25, 2015 from Willy Online Library.

Emaxsoftware (2007), ColdFusion MX Application Development. Retrieved June, 20, 2015 from http://www.emaxsoftware.com/general/Custom_ColdFusion_Programming

Find the best (2014), Compare Genealogy and Family Tree Software. Retrieved April 12, 2014, from <http://genealogy-software.findthebest.com>.

Geni (2011), Geni World Family Tree Grows To 60 Million People. Retrieved April 12, 2014, from <http://www.geni.com/corp/press/geni-world-family-tree-grows-to-60-million-people>.

Gensoftreviews (2011), Wikitree. Retrieved May 12, 2014, from <http://www.gensoftreviews.com/?p=1247>.

Hallogram (2008), ColdFusion MX. Retrieved June, 20, 2015 from <http://www.hallogram.com/coldfusion>.

Herrington, J. (2003), the PHP Scalability Myth. Retrieved June, 12, 2015 from http://www.onjava.com/pub/a/onjava/2003/10/15/php_scalability.html.

Kevin, Curran. (2012), Google+ vs. Facebook: The Comparison. **TELKOMNIKA (Telecommunication Computing, Electronics and Control)**, 10 (2), 379-388.

Kwak, H. Lee, C. Park, H. and Moon, S. (2010), what is Twitter, a social network or a news media? **In Proceedings of the 19th international conference on World Wide Web**, pp. 591-600.

Marion, A. and Omotayo, O. (2011) **Development of a Social Networking site with a Networked Library and Conference Chat** , Ogun State, Nigeria.

Mislove, A. E. (2009), **Online Social Networks: Measurement, Analysis, and Applications to Distributed Information Systems**. ProQuest.

My Heritage (2012), Introduce Records matching. Retrieved April 25, 2014, from **<http://blog.myheritage.com/2012/09/introducing-record-matching>**

Phil Inman, Sean Dodson (2004), "Genes reunited", The Guardian. Retrieved April 14, 2014, from **<http://www.theguardian.com/technology/2004/apr/29/media.newmedia..>**

Tamassia, (2013), Handbook of graph drawing and visualization. CRC press.

Wikipedia (2004), Family tree. Retrieved April 25, 2014, from **http://en.wikipedia.org/wiki/Family_tree**.

Wikipedia (2010), Tree (DATA structure. Retrieved May 15, 2015, from **http://en.wikipedia.org/wiki/Tree_%28data_structure%29**.

Wirth, N. (1986), **Algorithms and data structures**, New York: SpringerVerlag

Wikipedia (2014), My Heritage. Retrieved April 15, 2014, from **<http://en.wikipedia.org/wiki/MyHeritage>**.

Appendix A

إستبيان حول شبكات التواصل الاجتماعي التي تتضمن روابط النسب

في هذا الاستبيان نستطلع آراء مستخدمي مواقع التواصل الاجتماعي حول أهم الميزات والخدمات التي يرغبون في إضافتها إلى شبكات التواصل الاجتماعي والمتعلقة بروابط النسب (العائلة والأقارب). فيما يلي بعض الميزات المقترحة والتي نرغب في استطلاع آرائكم عنها.

جنس المستجيب لهذا الاستبيان: ذكر أنثى

المستوى التعليمي: جامعي غير جامعي

الفئة العمرية: أقل من 20 من 20 إلى 40 أكبر من 40

| الرقم | الميزة | مهم جدا | مهم | متوسط | غير مهم |
|-------|---|---------|-----|-------|---------|
| 1 | شجرة العائلة يمكن للمستخدم بناء شجرة العائلة الخاصة به ألياً من خلال الاستفادة من بيانات وسجلات حسابات مستخدمي مواقع التواصل الاجتماعي ودعوة أقاربه للانضمام إلى شجرة العائلة. | | | | |
| 2 | معرفة أصل وسلالة العائلة تمكن هذه الخاصية من معرفة أصل وسلالة العائلة الخاصة بالمستخدم من خلال الاستفادة من شجرة عائلته ومعرفة علاقاتها بالعائلات الأخرى: معرفة تاريخ العائلة و إنجازاتها من خلال المعلومات المحفوظة من قبل مؤسس هذه الشجرة. | | | | |
| 3 | الوصول إلى أعضاء العائلة بعد إن أصبح المستخدم جزءً من شجرة العائلة يمكن لأي عضو من أعضاء العائلة الوصول إلى حسابه ودعوة أعضاء العائلة ليكونوا في قائمة أصدقائه. | | | | |
| 4 | فصل الأصدقاء عن الأقارب في قائمة أصدقاء المستخدم يمكن لشبكة التواصل الاجتماعي التعرف بسهولة على أقاربه وعلى أصدقائه وعمل قائمتين: قائمة للأقارب وقائمة للأصدقاء دون الحاجة لتعريفهم من قبل المستخدم. | | | | |
| 5 | إشعارات الأقارب عند فتح المستخدم لموقع التواصل الاجتماعي يتلقى إشعارات حول أهم الأحداث والنشاطات التي قام بها أقاربه ويراعى في ذلك ترتيب الإشعارات حسب درجة القرابة: القرابة من الدرجة الأولى فالثانية وهكذا. | | | | |
| 6 | صندوق رسائل الأقارب فصل صندوق رسائل الأقارب عن صندوق رسائل الأصدقاء واستخدام وسائل | | | | |

| | | | | | |
|--|--|--|--|--|---|
| | | | | أكثر أماناً في صندوق رسائل الأقارب لضمان السرية. تحديث صندوق الأقارب بأهم الأحداث والمناسبات التي تخص شجرة العائلة وإرسالها إلى كل أعضاء العائلة. | |
| | | | | تجميع العائلات في عائلة واحدة تمكن هذه الخاصية من تجميع بيانات كل العائلات التي تتشابه في الأصل في شجرة واحدة لعمل شجرة موحدة داخل البلد الواحد. | 7 |
| | | | | توثيق الأقارب في شجرة العائلة ليس كل من يملك حساباً ضمن أسم العائلة بإمكانه الانضمام لشجرة العائلة، لذا يجب التأكد من بيانات أعضاء العائلة وتوثيق المستخدم قبل إضافته من خلال طلب الرقم الوطني أو ما شابه. ويتم التأكد من بياناته من مؤسس شجرة العائلة. | 8 |
| | | | | عمل شجرة عائلة متعاونة كل مستخدم ينضم إلى شجرة العائلة يتم إضافته إلى مجموعة العائلة بحيث يمكن تبادل معلومات العائلة، الاتصال بين أفراد العائلة والتعرف إلى تاريخ العائلة . | 9 |

* هل ترغب / ترغبني في إضافة ملاحظات أخرى:

.....

.....

.....

.....

نشكر لكم تعاونكم

Appendix B

استبيان حول موقع مجموعة العائلة

في هذا الاستبيان نستطلع آراء مستخدمي موقع مجموعة العائلة حول الخصائص والخدمات التي يقدمها الخصائص و التي تتعلق بروابط النسب (الاقارب والعائلة), فيما يلي بعض الخصائص والخدمات المقدمة والتي نرغب في استطلاع ارائكم عنها .

جنس المستجيب لهذا الاستبيان: ذكر أنثى

المستوى التعليمي: جامعي غير جامعي

الفئة العمرية: أقل من 20 من 20 الى 40 أكبر من 40

| الرقم | الميزة | ممتاز | جيد جدا | جيد | ضعيف |
|-------|---|-------|---------|-----|------|
| 1 | عرض شجرة العائلة يمكن لمستخدمي مواقع التواصل الاجتماعي عرض شجرة العائلة الخاصة بهم من خلال موقع العائلة والتعرف الى اصل وسلالة العائلة والوصول الى حسابات اعضاء العائلة من خلال النقر على اسمائهم في شجرة العائلة . | | | | |
| 2 | الانضمام لشجرة العائلة يمكن لمستخدمي مواقع التواصل الاجتماعي طلب الانضمام لموقع مجموعة العائلة ليصبح عضوا في المجموعة وجزءا من الشجرة. تمكن هذه الخاصية الأعضاء رؤية نشاطات العائلة المختلفة في المجموعة , والمشاركة فيها بالإضافة الى تعديل شجرة العائلة الخاصة بالمجموعة | | | | |
| 3 | تعديل شجرة العائلة تمكن هذه الخاصية اعضاء المجموعة من تعديل الشجرة وذلك بإضافة اقربائهم الى الشجرة, او حذف بعض منهم او التعديل على معلوماتهم وذلك من خلال طلب تعديل الشجرة من مؤسس المجموعة. | | | | |
| 4 | النشر على حائط المجموعة تمكن هذه الخاصية اعضاء مجموعة العائلة من النشر على حائط المجموعة. عندما يقوم احد اعضاء العائلة بنشر شيء على حائط العائلة, كل اعضاء العائلة سوف يرون المنشور ويتمكنون من التعليق عليه . | | | | |
| 5 | الخروج من مجموعة العائلة يمكن لأعضاء العائلة الخروج من المجموعة ولكن معلوماته سوف تبقى على شجرة العائلة. سوف يتحول عضو العائلة الى | | | | |

| | | | | | |
|--|--|--|--|--|---|
| | | | | مستخدم عادي: لن يتمكن من رؤية نشاطات العائلة او تعديل الشجرة. | |
| | | | | الموافقة على تعديل شجرة العائلة تمكن هذه الخاصية مؤسس مجموعة العائلة من الموافقة او رفض طلبات تعديل شجرة العائلة من قبل اعضاء العائلة | 6 |
| | | | | الموافقة على طلبات الانضمام للمجموعة تمكن هذه الخاصية مؤسس المجموعة من الموافقة او رفض طلبات الانضمام للمجموعة من قبل مستخدمي مواقع التواصل الاجتماعي | 7 |
| | | | | بناء مجموعة العائلة تمكن هذه الخاصية مستخدمي مواقع التواصل الاجتماعي من بناء المجموعة الخاصة بعائلاتهم وبناء الشجرة الخاصة بهم وإضافة اقرانهم. | 8 |
| | | | | تعديل شجرة العائلة من قبل مؤسس الشجرة تمكن هذه الخاصية ممن خلاله من التعديل على الشجرة وبنائها مباشرة. | 9 |

بعد استخدام الموقع واستطلاع اراء مستخدمي موقع مجموعة العائلة حول الخدمات التي يقدمها. سوف يتم استطلاع مستخدمي الموقع حول اهم الخصائص المتوفرة في الموقع والتي تهمهم كمستخدمين. فيما يلي بعض الخصائص والتي نرغب في استطلاع ارائكم عنها

| ضعيف | جيد | جيد جدا | ممتاز | الميزة |
|------|-----|---------|-------|--|
| | | | | سهل الاستخدام يمكن لمستخدمي شبكات التواصل الاجتماعي التعامل مع مجموعة العائلة بسهولة كونه يشبه تطبيق المجموعة الموجود في شبكات التواصل الاجتماعي. |
| | | | | مفتوح ومجاني للجميع موقع مجموعة العائلة مفتوح ومجاني للجميع فقط يتطلب وجود حساب على مواقع التواصل الاجتماعي. |
| | | | | سرعة الاداء موقع مجموعة العائلة سريع الاداء ويمكن مستخدميه من استخدام ميزاته بأقل وقت ممكن. |
| | | | | الدقة موقع مجموعة العائلة دقيق وخالي من الاخطاء |

بحث في شبكات التواصل الاجتماعي التي تتضمن روابط النسب

إعداد

عالية الحسيان

المشرف

الأستاذ الدكتور غيث عبدة

ملخص

في هذه الرسالة نقترح تطبيقا لربط شبكات التواصل الاجتماعي مع خدمات روابط النسب. التطبيق يدعى مجموعة العائلة حيث يربط مستخدمي شبكات التواصل الاجتماعي من خلال شجرة العائلة الخاصة بهم. وقد خلصت الدراسة إلى عدد من النتائج والتوصيات من أهمها: تمثيل أهم خدمات روابط النسب التي تم استطلاعها في استبيان مسبق في هذا التطبيق والذي تم تصميمه وفقا لإحتياجات المستخدمين وتم تجريبه من قبل عدد من المتطوعين لتجربة هذه الخدمات. وقد تلخصت نتائج هذه التجربة بأن الخدمات تم تمثيلها بشكل كامل في هذا التطبيق كما تم اختبار شجرة العائلة وتبين تمكنها من رسم شجرة ذات اعداد كبيرة في زمن قصير.